THE DEVELOPMENT OF SCADA CONTROL AND REMOTE ACCESS FOR THE INDLEBE RADIO TELESCOPE

Submitted to the Department of Electronic Engineering in the Faculty of Engineering and the Built Environment at the Durban University of Technology, in fulfillment of the academic requirements for the Degree

Master of Engineering (Electronic)

by

Ajith Deoduth Dhaniram

October 2015

Mr GP Janse Van Vuuren	Date
Supervisor	
Dr P Govender	Date
Co-supervisor	

DECLARATION OF ORIGINALITY

I hereby declare that the information presented within this thesis is my original work and has not been published or submitted for publication. Any information derived from others and used in my thesis, has been acknowledged in the text and referenced according to the Durban University of Technology requirements.

_____ _____

Mr A D Dhaniram Date

19201056

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude and appreciation to the following persons:

1. Gary van Vuuren: for his support, guidance, time, advise and going the extra mile as my supervisor for this project.
2. Bradley Le Roux: for assisting with technical advise and support on the Adroit system.
3. Heinz Muller: for advising on the implementation of single SMS functionality on the GSM Commander system, when a behavior is triggered.
4. Jonathan Quick: for sharing the system design and functionality of the HartRAO radio telescope.
5. Shoanaha Dhaniram: my wife for her support and motivation.

ABSTRACT

The proposed supervisory control and data acquisition solution is intended to gather data from all sub-systems and provide control commands related to the Indlebe Radio Telescope. Currently the control commands are executed from the command line prompt of the Skypipe software. These control commands are used to change the elevation angle of the antenna.

The supervisory control and data acquisition system will be interfaced to sub-systems namely; a programmable logic controller, a weather station, an uninterruptible power supply and a camera. It will be used to manually or automatically control the elevation angle of the antenna, includes a menu structure that allows for easy navigation to the sub-systems and allows for trending, alarming, logging and monitoring of all system parameters. The proposed system will mitigate the lack of information on the existing system.

A global system for mobile communication unit has also been installed to monitor the temperature within the Indlebe control room, detect a power failure and communicate this information to supervisors, using its short message service option.

Implementing a solution of this nature means that all data from the various sub-systems are brought together, giving a single platform to monitor data and provide manual and automatic control functionality. Problem solving, understanding and maintenance of the system will also become easier.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF PHOTOGRAPHS

LIST OF TABLES

ABBREVIATIONS AND SYMBOLS

A	-	Ampere
AC	-	Alternating current
Ah	-	Ampere hour
CPU	-	Central processing unit
CSV	-	Comma separated value
DC	-	Direct current
DIT	-	Durban Institute of Technology
DUT	-	Durban University of Technology
EBE	-	Engineering and the Built Environment
GPRS	-	General packet radio service
GSM	-	Global system for mobile
HartRAO	-	Hartebeesthoek Radio Astronomy Observatory
HMI	-	Human machine interface
HTML	-	Hypertext markup language
Hz	-	Hertz
ICR	-	Indlebe control room
IP	-	Internet protocol
IRT	-	Indlebe Radio Telescope
kW	-	Kilowatt
LAN	-	Local area network
LOFAR	-	Low Frequency Array
m	-	Meter

mA	-	Milliampere
MAC	-	Monitoring and control
MAC	-	Media access control
MITRA	-	Multi-Frequency Interferometer Telescope for Radio Astronomy
MHz	-	Mega hertz
NCCS	-	New computer control system
PC	-	Personal computer
PCI	-	Peripheral component interconnect
PID	-	Proportional integral derivative
PLC	-	Programmable logic controller
PMAC	-	Programmable multi axis controller
PSU	-	Power supply unit
PVC	-	Polyvinyl chloride
RRI	-	Raman Research Institute
SCADA	-	Supervisory control and data acquisition
SIM	-	Subscriber identification module
SMS	-	Short message service
UI	-	Universal interface
UPS	-	Uninterruptible power supply
URL	-	Uniform resource identifier
USB	-	Universal serial bus
V	-	Voltage
VA	-	Voltage ampere
VB	-	Visual basic

VDC	-	Voltage direct current
VSD	-	Variable speed drive
W	-	Watt
%	-	Percentage
°C	-	Degree Celsius
°	-	Degree

CHAPTER 1

INTRODUCTION

1.1 Introduction

The Durban University of Technology (DUT) is a result of the merger in April 2002 of two prestigious technikons, ML Sultan and Technikon Natal. Initially it was named the Durban Institute of Technology (DIT) and later became the DUT in line with the rest of the universities of technology. DUT is located in the city of Durban, on the east coast of KwaZulu Natal. DUT, a member of the International Association of Universities, is a multi-campus university of technology at the cutting edge of higher education, technology training and research.

The Department of Electronic Engineering is one of the departments within the faculty Engineering and the Built Environment (EBE) of DUT. In 2006 the department embarked on a project to design, construct and test a radio telescope operating at the hydrogen line frequency of 1420 MHz. The Zulu phrase “indlebe zikhayi langa” literally means those whose ears glow in the sun. This phrase gave rise to the project name Indlebe Radio Telescope (IRT).

The purpose of the IRT is to provide a real world platform for project work for students within the Department of Electronic Engineering. Although science was not an initial goal of the IRT, it has demonstrated the potential to conduct useful astronomical observations.

1.2 Background

Radio astronomy is a subfield of astronomy that studies astronomical objects at radio frequencies.

These objects are different sources of radio emissions, typically the Milky Way, Galaxies, Pulsars, Masers and the Sun. These sources are detected with a radio telescope.

At the Steve Biko Campus of DUT, the IRT (refer to **Photograph 1.1**) is mounted at the lower level on a concrete beam running between two five-level tower blocks. The radio telescope uses a 5 m parabolic reflector antenna. The two tower blocks are orientated at 17,52° northeast. The antenna has a 40° elevation control, where the horizontal position is 0°, the northern position is 20° and the southern position is -20°.

Photograph 1.1 – A view of the Indlebe Radio Telescope at DUT

The construction of the IRT at this location meant little exposure to wind loading, that resulted in a cost effective design. The build being a very lightweight structure that is well balanced on its center of gravity, to a smaller drive motor, gearbox and variable speed drive (VSD). Due to the

lightweight design, the mechanical system can easily be damaged and included a high degree of backlash. On implementing a control solution, the mechanical constraints had to be taken into consideration to achieve accurate positioning of the antenna.

1.3 Problem

The existing standalone system comprises of a data capture computer that also runs the SkyPipe software and is only available from the Indlebe control room (ICR) at DUT. The SkyPipe system is used to monitor celestial source positions relative to the location of the IRT on earth's surface and thereby determine the elevation angle of the antenna to successfully measure the celestial source. The user will then input this elevation angle into the antenna positioning computer using a command line instruction. The antenna will then start to move, however there is no way of visually checking that the antenna is moving in the correct direction and has reached the desired elevation angle. As potential sources appear in the telescope beam throughout a 24 hour period, continuous monitoring and repositioning of the telescope is required.

If there are any faults in the control system, the current system will not be able to determine such a fault or log the data. Information pertaining to the field of astronomy that this system delivers is not easily accessible. Monitoring and control of the system is not possible or very limited.

1.4 Aims

One of the most important aims of the IRT is to increase the awareness and interest in the field of

science and technology. As part of this it is intended to make the data available to the general public, in real time, and in particular to schools throughout the country (and internationally). Achieving this will be via the internet with a web server.

In order to bring all control aspects of the IRT together, a supervisory and data acquisition (SCADA) system is to be implemented. The SCADA system will gather data from all of the sub-systems and provide control commands, as well as provide a host of other features common on commercial SCADA packages.

The aims of this project are:

- To investigate the control strategies of some other telescopes.
- To consolidate all sub-systems within one control environment.
- To make the data that the radio telescope gathers available via the Internet.

1.5 Objectives

The objectives of this project are:

- To implement a SCADA control system. The SCADA system will receive data from all the sub-systems of the IRT.
- To implement a wireless weather station to provide real time weather data. The SCADA system will monitor and trend this information.

- To implement a remote controlled industrial camera. This will provide a real time visual image of the telescope.
- To implement an uninterruptible power supply (UPS) alarm and status monitoring. The UPS is used to electrically protect the equipment in the ICR from mains failure or disturbances. The SCADA system will monitor and log this information.
- To interface with a programmable logic controller (PLC) that is used to control the movement of the IRT.

CHAPTER 2

SUPERVISORY CONTROL AND DATA AQUISITION

2.1 Introduction

There are many radio astronomy observatories throughout the world and even in space. These include single antenna and array applications. During the course of this project the control methods of the radio telescope being built at the Raman Research Institute (RRI), the Low Frequency Array (LOFAR) radio telescope and the radio telescope at the Hartebeesthoek Radio Astronomy Observatory (HartRAO) were investigated.

2.1.1 The Raman Research Institute (RRI)

A 12 m radio telescope antenna, which operates in the frequency range 0,5 to 8 GHz, is being built at the RRI in India. A programmable multi axis controller (PMAC) is being used as the position controller. PMAC is a multi processing system, dedicated for handling different control tasks. The PMAC also comes with encoder interfaces, real time clock, analog to digital and digital to analog input/output cards, operating system and advanced proportional integral derivative (PID) control algorithms. The system uses brushless direct current (DC) motors to move the antenna and DC drives for velocity control. There are two 17-bit absolute single turn encoders, providing position feedback for azimuth and elevation axes. The control system uses a method called backtorquing to minimize the effects of backlash by controlling two motors with an offset in their command velocities in the same axis, thereby minimizing the backlash in the gears [1].

A personal computer on the observers desk sends source coordinates to a primary control path known as the PMAC control personal computer. The source coordinates are also sent to a secondary control path known as the Linux based personal computer which provides redundancy and enhanced reliability. The PMAC personal computer takes these coordinates and passes them the PMAC program motion control variables. The astronomical coordinate system conversions are implemented inside PMAC, using the C programming language. The program continuously calculates the expected azimuth and elevation angle from the source coordinates and passes it on as position commands to the PMAC servo channels [1].

The Linux personal computer interfaces to the control system hardware through commercially available peripheral component interconnect (PCI) based data acquisition cards and digital input/output cards. The data acquisition card interfaces with position encoders for acquiring position feedback and also commands the motor drives for motion control. The digital input/output card interfaces with the safety interlocks and focus control systems. The control system uses both the PMAC and Linux options, has been successfully tested using a test rig under simulated conditions similar to an actual radio telescope system. The 12 m radio telescope fabrication work is in the final stages of completion and the next step is to try out the control system developed, with the actual telescope [1].

2.1.2 The Low Frequency Array (LOFAR) radio telescope

The LOFAR radio telescope is a wide area sensor network covering an area of 350 square kms, located in the northern part of the Netherlands. Distributed over this area are 77 field stations, where each field station contains 100 antennas for astronomy observations and 10 sensors for

geographical and high precision agricultural measurements. Being a phased-array radio telescope, it operates over a very low frequency range from 30 to 240 MHz. There are two sets of antennas installed where the one set is sensitive to the 10 to 80 MHz range and the other set is sensitive to the 110 to 240 MHz range. At the station sites, the individual antenna signals are filtered in both the RF and the digital domain after which all antenna signals are combined in spatial sensitive signals that are sent to the central site for further processing [2].

The monitoring and control (MAC) system of LOFAR is distributed throughout the station sites and central control site. MAC is used to monitor and control the execution of observations to all systems in the distributed area. There are six workstation terminals located at the LOFAR operations central control site, used to run the observation models. The MAC navigator is the graphical user interface application that is provided to all users of the MAC system. The navigator is designed in a tree browser structure and the operator can select a specific station in the different site locations [2].

The system is used to gather information from all equipment, provide a drill down navigation option to monitor details of field instruments being used and provide the required distributed database and user interface functionality. Using the navigation option the operator can monitor or change parameters to trend plots, trigger levels of control actions, set new alarm levels, etc. Other features include logging, archiving, reporting, alerts, access control and script control. The presented information and available pages in the navigator depends on the user's role, in other words astronomer or operator. Due to the LOFAR system being so large, complex and serving multiple users, it is important that the user interface system is simple, has a unified look and feel, minimum maintenance and a single graphical interface [2].

2.1.3 The Hartebeesthoek Radio Astronomy Observatory (HartRAO)

The HartRAO is located in a valley in the Magaliesberg hills, 50 km north-west of Johannesburg, in the province of Gauteng, South Africa. The main telescope has a reflecting surface diameter of 26 m and operates at frequencies of 1,66 to 23 GHz. All observing is controlled by a computer system. The new computer control system (NCCS), based on networked pc's use the Linux operating system. The software application used is based on object oriented device control software.

In a telephonic conversation on the 15 October 2014, the Program Leader, Dr Jonathan Quick described the system design and functionality of the 26 m antenna at HartRAO. Each axis is driven by a pair of 9 kW three phase electric motors through reduction gearboxes. The motors are driven by drive controllers and work in conjunction for high speed driving, and opposing each other in an anti-backlash arrangement at low speeds. The 26 m telescope operates under direct computer control by the NCCS. A Eurotherm PLC is used in the control architecture. The system uses two 19-bit digital encoders to determine the position of the antenna. The position of the antenna is compared to a setpoint and an error is calculated. The system will move the antenna position until the error is less than or equal to $0{,}001^{th}$ of a degree. In addition the system is also designed for tracking celestial sources. For a single source the NCCS system inputs a setpoint and the antenna will move to this setpoint and then tracks the source. For multiple sources, the NCCS system executes a scheduled program based on the celestial sources to be observed. The system prioritizes the sources in terms of appearance above the horizon and determines the coordinates of the source. Each source is tracked and measured for a period of 10 minutes. The

computer system also has a weather station associated with it. The system has been designed to park the antenna during severe weather conditions.

2.2 The Indlebe Radio Telescope (IRT) system

The control and monitoring systems used by the three observatories is based on customized systems namely; PMAC, MAC and NCCS. LOFAR has non steerable antennas as compared to RRI and HartRAO. The systems used at RRI and HartRAO are interfaced to onboard cards or a programmable logic controller that is used to position the antenna. In each case the antennas are moved by motors and the speeds of these motors are controlled using drive controllers. At HartRAO, the antenna is automatically parked when excessive wind speeds are detected using their weather station. The control and monitoring system used by IRT is based on industry best practices that includes a commercially available SCADA system, a PLC and a VSD. The weather station at DUT is monitored using the SCADA system. There is no interlocking functionality in the control of the antenna for varying weather conditions. RRI, LOFAR and HartRAO are proper research telescopes as compared to IRT. It is for this reason that the capital investment of these telescopes far exceeds that of the IRT.

The software code used to control the position of the antenna at RRI is written using the C programming language and requires specialized skills which can be disadvantageous. The software code used to control the position of the antenna at IRT is written using the VB programming language and is well understood by most programmers. HartRAO has the highest resolution encoders and is therefore the most accurate. The antennas at HartRAO are controlled

to 1 pulse per 0,001°, whilst RRI is controlled to 1 pulse per 0,0027° and IRT is controlled to 1 pulse per 0,352°.

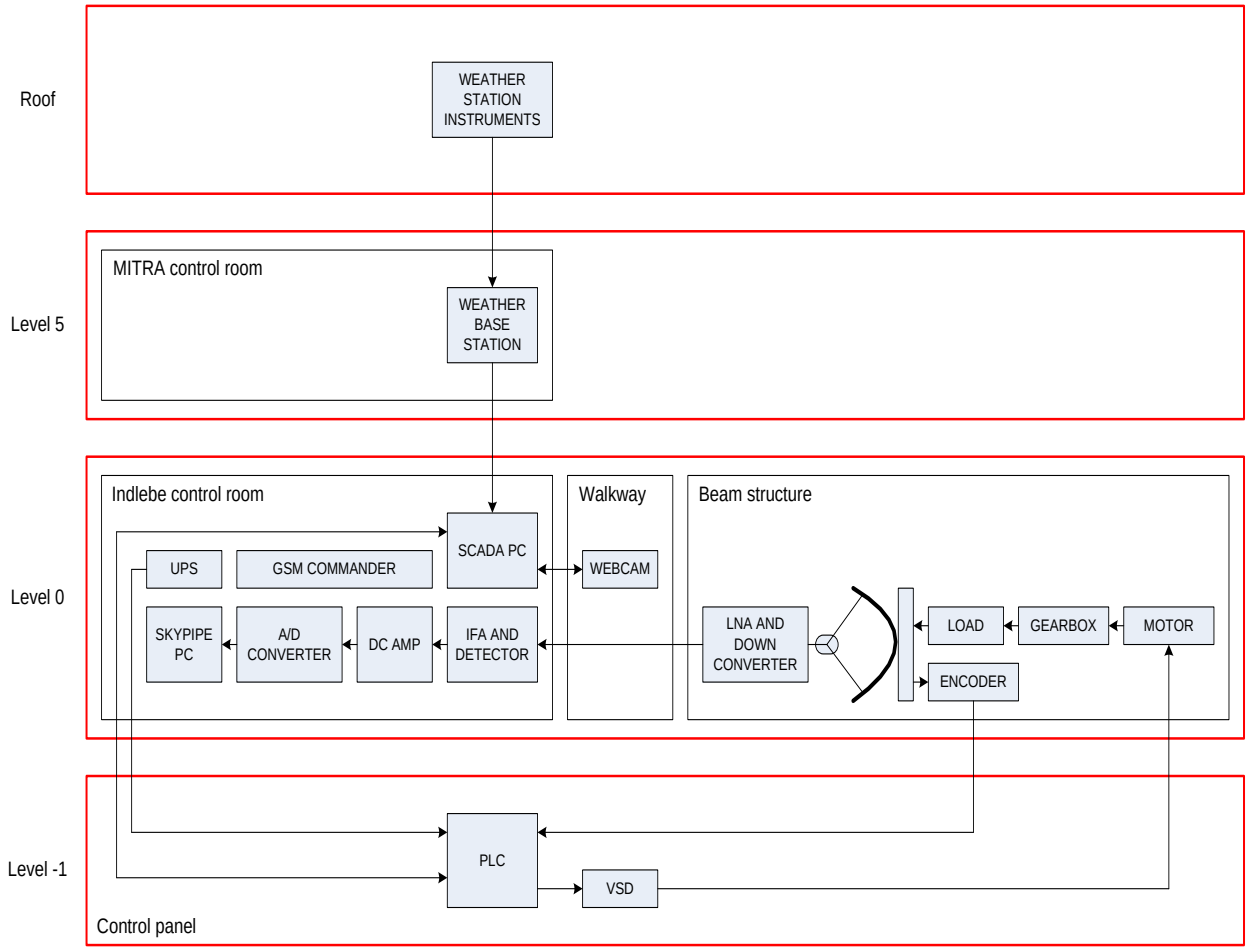


Figure 2.1 – Block diagram of the proposed layout.

The SCADA system is interfaced to a PLC that is used to position the antenna. The antenna is moved by a motor and gearbox arrangement and the speed of the motor is controlled using a VSD. The SCADA system was donated by Adroit Technologies to DUT in an effort to skill future engineers by providing them with the tools and knowledge in the field of control engineering (version: 7.0.2.1, serial number: 8780, scan points: 750 and remote nodes: 2). The block diagram shown in **Figure 2.1** is proposed for the IRT system. Several different sub-systems

need to be catered for namely; a PLC, a weather station, a UPS, a global system for mobile (GSM) commander and a camera.

2.3 Supervisory control and data acquisition system

Historically, a process used to be represented on a panel with pushbuttons, indicator lights, pen recorders, panel meters, level indicator displays, printers, etc. Today, an operator has a user friendly graphical interface where the overall view of the process is displayed on a computer screen.

A SCADA system is not a control level system, but rather focuses on the supervisory level. It is purely a software package that is positioned on top of hardware to which it is interfaced, in general with a PLC or other commercial hardware modules. These include network switches, camera's, etc [3],[4]. The SCADA hardware is the computer system and the software is the application that is installed on the computer system. The SCADA software is a means of real time graphically monitoring, recording and controlling a process. **Figure 2.2** illustrates a basic SCADA system, where the system is interfaced to a PLC for reading and writing data, using the standard Ethernet communication protocol. The PLC sends and receives information from field devices in local or remote process locations. These devices include; pumps, drives, sensors, and so on.

Since 1992, Adroit Technologies, a South Africa company, have been developing SCADA software solutions for the industrial automation and process control industries. Adroit have donated many SCADA systems to a variety of institutions, including DUT. The donation is to

help ensure that these institutions of learning are equipping the future engineers with the skill-sets required in a competitive market [5]. An Adroit SCADA system has been donated for the purpose of implementation on the IRT project.

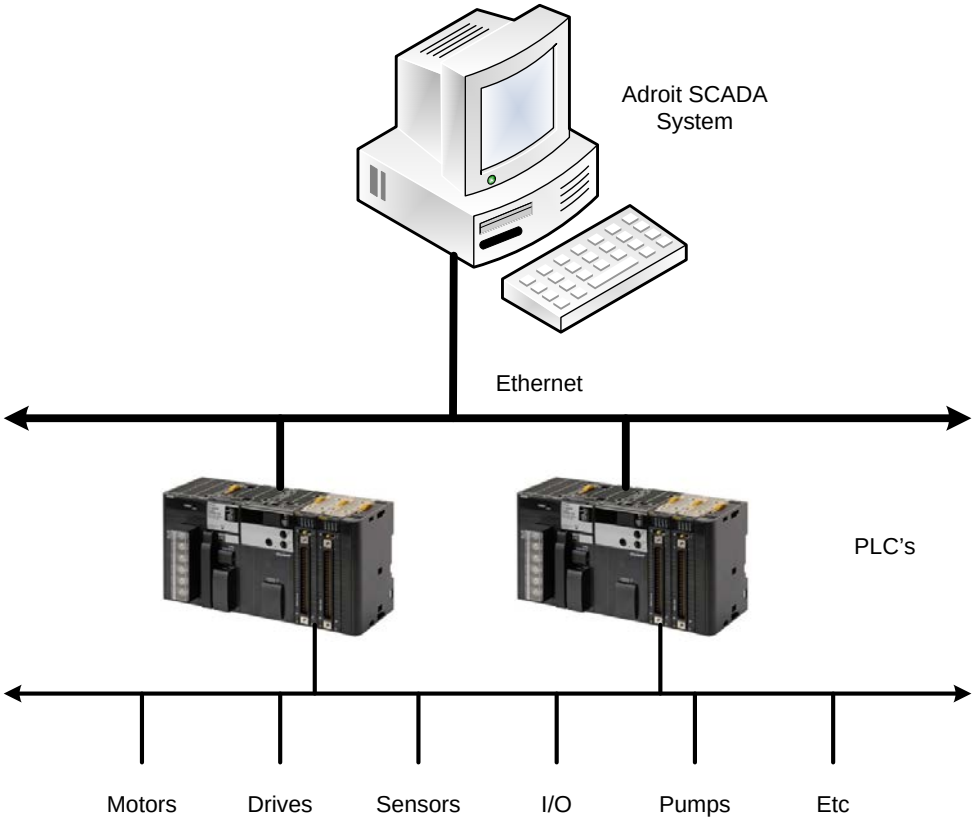


Figure 2.2 – Basic SCADA System

Other features available to the operator include; the ability to navigate to other areas of the process, drill down into details of the equipment used, graphical objects (motors, pushbuttons, level displays, etc) are used to interact with the process, and so on. Trending and data logging of values such as temperatures, liquid levels, production rates and motor speeds can be recorded and viewed over a specific period of time using a graphical trend window. Alarms and events are logged and can be viewed in real time or as historical information. SCADA software have

become so advanced that they can perform recipe functions, interlocking, scripting, hot standby/redundancy, communicate with other products using open communication protocols, short message service (SMS) functions, online configuration capabilities, and so on. The Adroit SCADA system includes multiple software namely; Adroit agent server, Adroit classic user interface, Adroit setup, Adroit script editor and an OMRON Ethernet protocol driver that is used for development, runtime and interfacing to other systems or devices. For the IRT system the type of software and driver installed and its purpose is as follows:

- Adroit agent server - is the real time database/engine used for logging or alarming or tags and scanning protocol drivers.
- Adroit classic user interface - the user interface software is used to build the mimics or displays.
- Adroit setup - the agent server application software is used to setup and configure the system.
- Adroit script editor – this is used to write script functions in visual basic or java.
- OMRON Ethernet protocol driver - the front end device drivers for communication to a PLC.

The system also includes a physical dongle that is connected to the universal serial bus (USB) port of the computer. The dongle is a license key that is based on the number of scan points and the number of allowable concurrent connections, authorizing the use of the Adroit system. **Figure 2.3** illustrates a block diagram of the Adroit SCADA system software and hardware. The Adroit software installation procedure can be referenced as per **Appendix A**. The Adroit SCADA system architecture as illustrated in **Figure 2.1**, includes different sub-systems that have been

integrated to the SCADA system. These sub-systems that are integrated into the SCADA system includes; a PLC, a weather station, a UPS and a camera. The sub-system that is not integrated into the SCADA system is the GSM commander.

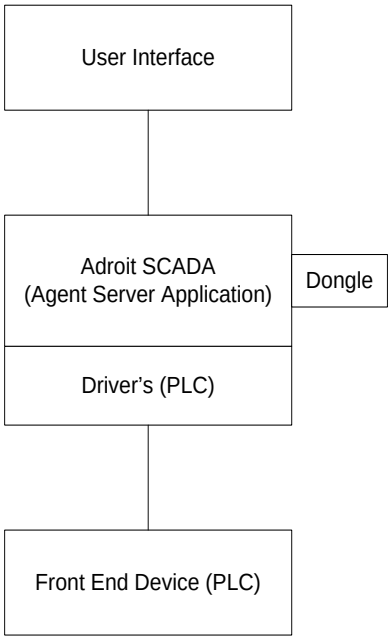


Figure 2.3 – Adroit SCADA system software and hardware

The SkyPipe computer and the SCADA computer are separate standalone systems. The output data from the SkyPipe computer is not described as it was not included in the scope of this project.

2.4 Sub-systems description

The sub-systems perform specific functions in the overall design and each will be described in detail below.

2.4.1 The programmable logic controller

The PLC is a critical component in the system that receives setpoints and control commands from the SCADA system and the human machine interface (HMI) system to change the elevation angle of the antenna. The PLC uses this information, with other interlocking variables and actual position feedback to safely and accurately move the antenna to the desired setpoint.

2.4.2 The weather station

The weather station is used to measure the local pressure, rainfall, temperature, humidity, wind direction and wind speed.

2.4.3 The uninterruptible power supply

The UPS is used to protect the data capture computer, the receiver front-end and back-end, in the event of a mains failure or disturbance. Monitoring the status and mode of operation of the UPS is important. The UPS alarm signals are connected to the PLC system and this data is transferred to the SCADA system using the PLC communication link.

2.4.4 The camera

The remote camera is used for real time visualization of the antenna position that is located outside the control room. The SCADA system displays this information on the camera screen.

2.4.5 The global system for mobile commander

The GSM Commander, although not interfaced to the SCADA system, is used to monitor the mains supply and control room temperature. Mains failure and high temperature alarms are communicated to DUT personnel via the SMS option.

Each of these sub-systems is described in more detail in **Chapter 3.** The SCADA system displays the information of the PLC, weather station and the uninterruptible power supply on the main overview screen.

2.5 Network description

Protocols used within SCADA communications are; modbus protocol, distributed network protocol, profibus protocol, foundation fieldbus, modbus plus protocol, data highway plus protocol, transport control protocol and internet protocol [6]. The various components of the system integrate to the SCADA system using two communication protocols i.e. USB and Ethernet. USB was designed to standardize the connection of computer peripherals (including keyboards, pointing devices, digital cameras, printers, portable media players, disk drives and network adaptors) to personal computers, both to communicate and to supply electric power. The Ethernet standards comprise several wiring and signaling variants. The original Ethernet used coaxial cable as the shared medium. Later the coaxial cable was replaced with twisted pair and fiber optic links in conjunction with hubs or switches. Data rates were periodically increased from the original 10 megabits per second to 100 gigabits per second [7].

The weather station communicates to the SCADA system using the USB communication protocol that has been developed to provide a fast, flexible method of attaching up to 127 peripheral devices to a computer. The USB provides a connection format designed to replace the systems traditional serial and parallel port connections [8]. USB works very much like the current plug-and-play, allowing you connect or disconnect devices without powering down or reconfiguring the computer. When you add or remove a new device , the system automatically detects the change, then loads or unloads the appropriate driver [9].

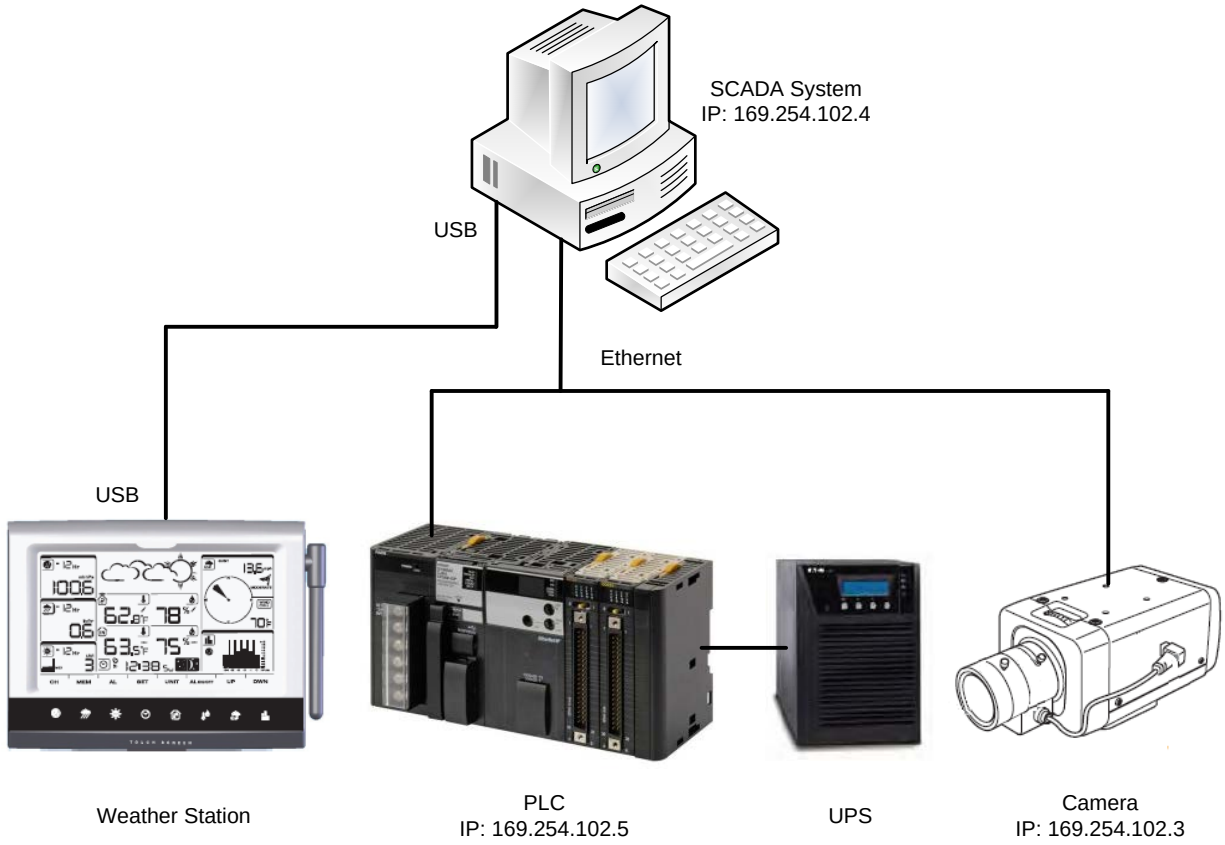


Figure 2.4 – SCADA network protocols and IP addresses

The PLC, camera and SCADA communicate on the Ethernet protocol. They all have internet protocol (IP) addresses that are in the same range. The UPS alarm signals are connected from a

relay interface card to the PLC using a multi-core cable. The SCADA network protocol is illustrated in **Figure 2.4**. A network is two or more computer systems or devices which are linked together by some form of transmission medium that enables them to share information. A network will provide services such as sharing of files and folders, network printing and database applications. An important aspect of networking is security. Simple network systems may use a password to control access to resources, while the more sophisticated systems control access by a user name. User names are an important part of network security and many network systems deny access until the user has provided a valid user name and password [10]. The SCADA system has been configured with a user name and password security requirement.

2.6 Design layout

The layout of the SCADA includes a navigation header and a main overview screen. The navigation header allows for navigation to six other screens namely; camera, source, alarms, events, trends and back to the main screen. Navigation is an effective concept that prevents all information being displayed on one screen but rather helps breakdown the entire project into different areas with specific focus on information being displayed for that area. The header is common to all screens therefore allowing navigation to and from any area.

The main overview screen displays information pertaining to the weather station, UPS, PLC, an angle converter and a mechanical assembly of the antenna. The trend screen has an option to select trends for the weather station, UPS, and PLC. Each trend option has predefined tags setup for trending. The trend selections available are for the weather station, UPS data, PLC data and the six sources. **Appendix B** can be referenced for the different screen layouts.

2.7 Design functionality

The elevation position of the antenna can be controlled either manually or automatically. The selection is done using a selector switch on the main overview screen. The software code is written in the visual basic (VB) language [11] used by the Adroit SCADA system. To operate the system manually, the user enters a position setpoint between a maximum (+20°) and minimum (-20°) limit. Once a valid setpoint value has been entered the antenna will move from its current position to the new setpoint position, controlled to within an accuracy of 0,35°.

The encoder that was chosen to achieve this tolerance is the Omron (E6C3), which is a typical absolute encoder that has a resolution of 1024 positions per revolution [12]. The resolution is also equivalent to 2,84 pulses per degree, which is further converted to 1 pulse per 0,352°. If another source has to be measured then the user must enter a new setpoint.

To operate the system automatically, the user has two options to setup the system to measure celestial sources. The first option of measuring multiple sources is achieved by entering the required data into the system and enabling the sources. The position setpoint must be between a maximum (+20°) and minimum (-20°) limit. The system will automatically move the antenna to the position setpoints as per a priority selection of the sources. The second option is to measure a single source from a maximum (+20°) position to the minimum (-20°) position, with a 1° position change for every 24 hours. On completion of this cycle, the source is automatically de-selected. The script for the manual and automatic control and its functionality can be referenced as per **Appendix C**.

The save option is used to save the automatic control parameters captured for the automatic control. In the event of a power failure these parameters are not lost. The configuration of the save option can be referenced as per **Appendix D**. The mechanical assembly of the antenna as displayed on the main overview screen gives a true representation of the actual position of the antenna. As the antenna moves from one position to the next the display changes its position accordingly.

The angle converter is a tool that was developed to convert the antenna elevation angle of +70° to 110°, as an interpreted elevation angle of +20° to -20°. **Figure 2.5** shows the antenna elevation angle conversion. The script for the angle converter can be referenced as per **Appendix E**.

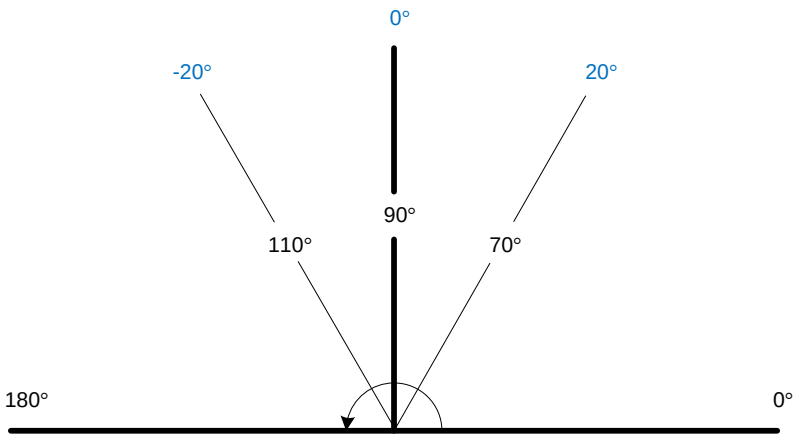


Figure 2.5 – Antenna elevation angle conversion

The Adroit SCADA software is an off the shelf product that does not require specialized skills to install, develop and maintain a system that incorporates this software.

CHAPTER 3

SUB-SYSTEMS

3.1 Introduction

The SCADA system is interfaced to sub-systems using different communication protocols, and gathers data from the sub-systems for graphical display, monitoring, recording and control purposes. The sub-systems include; a PLC, a weather station, a UPS, a GSM Commander and a camera. The SCADA and sub-systems is shown **Figure 3.1**.

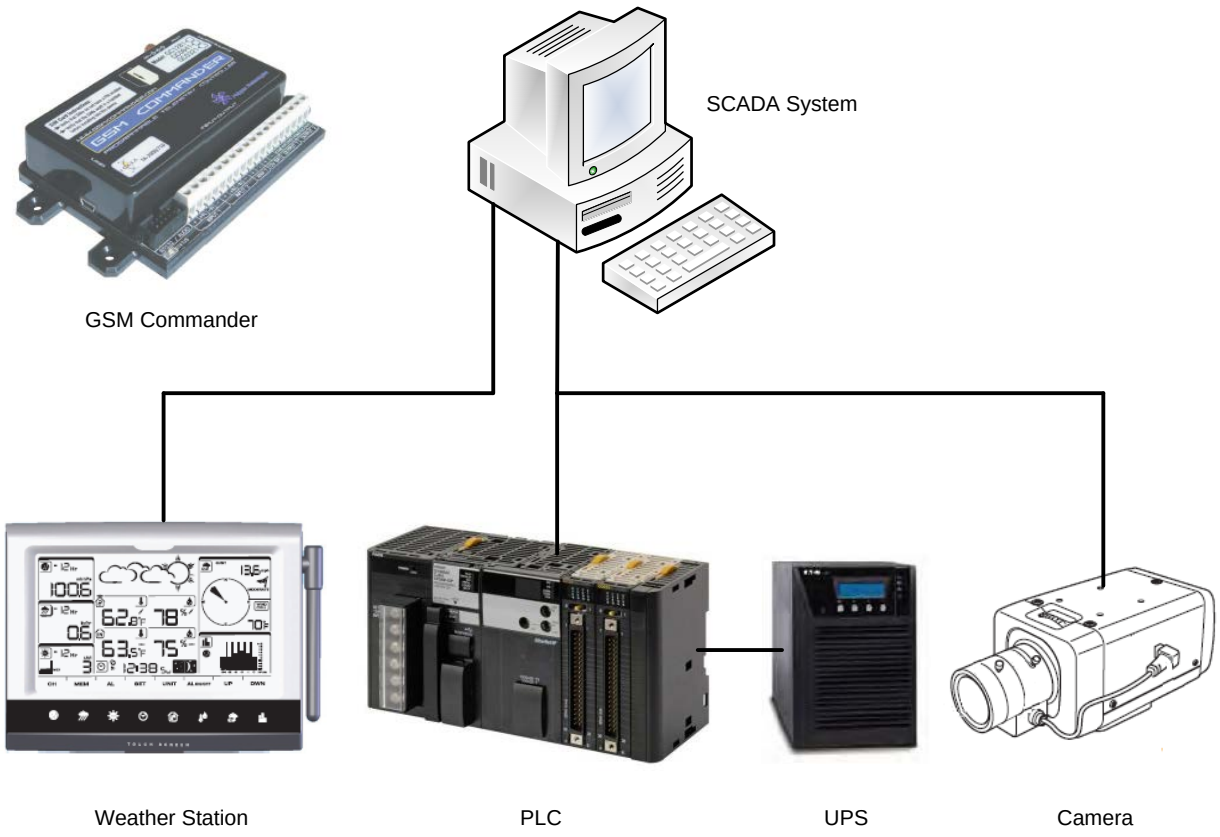


Figure 3.1 – SCADA sub-system block diagram

3.2 Programmable logic controller

3.2.1 Introduction

A PLC is a computer based device that uses a central processing unit, memory and other peripherals to execute a software program to control machinery and processes.

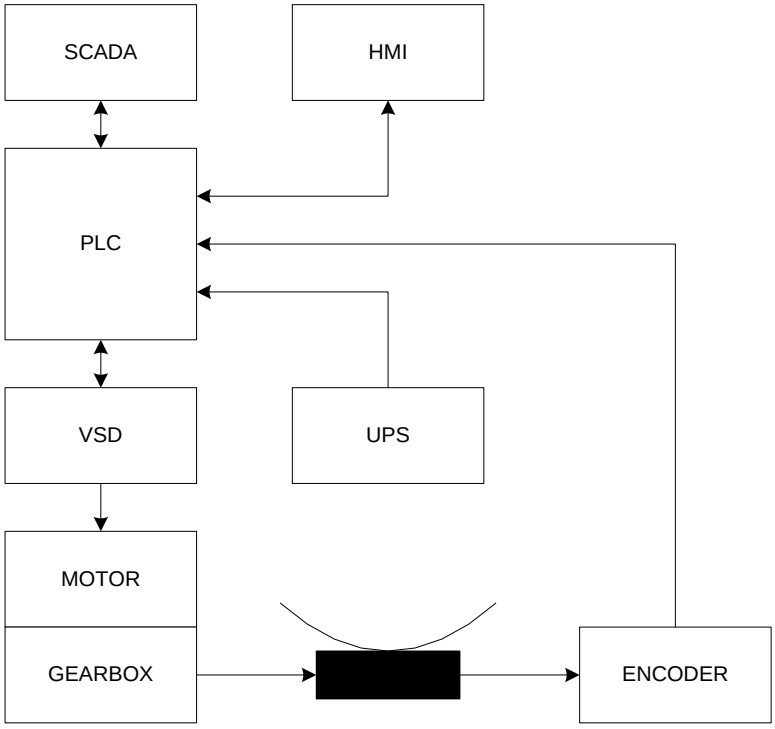


Figure 3.2 – PLC interfacing systems

The hardware used between various manufacturers differs in size, design, cost and specifications. Most PLC's conform to standard communication protocols namely; Ethernet, profibus, profinet, and so on. The PLC uses these protocols to interface to other systems or devices such as a SCADA, a VSD, a HMI, encoders, and so on. The software used by a PLC is developed against a

functional specification and programmed into it using a programming device. The basic block diagram in **Figure 3.2**, illustrates the PLC and its interfacing systems.

3.2.2 System functionality

The PLC, HMI and a VSD are installed in a panel located on level -1. The PLC hardware comprises of a central processing unit (CPU), type CPU11-ETN; an Ethernet module, type CJ1W-ETN2; two input modules, type CJ1W-ID211 and one output module, type CJ1W-OD211. A physical connection is established between the PLC and SCADA using an industrial Ethernet cable. Data is transferred between these two systems using the Ethernet protocol.

The system has been designed with four control methods namely; a jog north and south option from the HMI, a jog north and south option from the SCADA, a manual setpoint option from the SCADA and an automatic setpoint option from the SCADA. A key switch mounted in the panel is used to select between the HMI and SCADA control options.

The PLC is a critical component in the overall system architecture. Position setpoints or jog commands from either the SCADA system or the local HMI system is sent to the PLC. The PLC then transfers this setpoint and other control commands to the VSD, which controls the speed of the motor to move the antenna. The antenna position can be controlled in a north or south direction. Connected to the antenna is an absolute encoder (1024 pulses per revolution) that is used to measure the angle of the antenna. The PLC uses this variable to determine when the antenna reaches the desired position setpoint before stopping the motor. In all modes, the drive

runs the motor at 60 Hz for absolute antenna angle errors greater than 3° and when the antenna angle error is less than or equal to 3°, the drive runs the motor at 6 Hz.

When the antenna reaches the target of 0,35° from the setpoint, the motor is stopped for a period of 20 seconds, known as the calming period. Once the calming period ends the angle error is rechecked. If the antenna angle error is greater than 0,35° then the antenna is moved again to setpoint. When the antenna reaches setpoint, the motor is stopped for another 20 seconds. Once the calming period ends, the angle error is rechecked. If the antenna angle error is less than or equal to 0,35°, no further movements take place [13]. The antenna will reach its setpoint position within 3 attempts. The calming period allows the oscillations to decay before measurements can take place. The oscillation is due to the flexibility of the antenna supporting frame that results in the antenna moving past the position setpoint therefore the system will reduce the motor speed so as to minimize this oscillation.

The antenna position (actual) is the angle of the antenna in degrees. Because of the location of the telescope the only possible elevation angle is between 70° and 110°, where this range is interpreted as +20° to -20°. For a positive value the antenna bearing is north and for a negative value the antenna bearing is south. The antenna position (setpoint) is the angle setpoint of the antenna in degrees. Changes to this setpoint can only be done when the system is set to manual mode. The antenna position error is the error between the actual antenna position and the desired setpoint. The VSD continues moving the antenna until it is within 0.35° from the setpoint.

The north and south angle limit, is the maximum software north limit of the antenna in degrees (+20°) and the maximum software south angle limit of the antenna in degrees (-20°). The antenna

is not to be allowed to exceed this position and the motor will be stopped when this position is reached. There is also an emergency north and south limit switch that is mounted on the antenna structure at +25° and -25°. These limits are also hardwired to the variable speed drive to protect the antenna from being driven past these fixed positions. If the emergency limit switch position is reached, the motor is stopped. To recover from this position the antenna is moved in the opposite direction.

The operator panel (auto) selection is when the key switch mounted in the panel is selected to SCADA control. The “SCADA key” text is displayed and green status indicator is on. If this option is not selected, then the status indicator turns to red and the text disappears. The operator panel (local) selection is when the key switch mounted in the panel is selected to HMI control. The “HMI key” text is displayed and the green status indicator is on. If this option is not selected, then the status indicator turns to red and the text disappears.

The variable speed drive run/stop indicates when the PLC is sending run commands to the VSD. The “Running” text is displayed and the green status indicator is on. The “Stopped” text is displayed and the red status indicator is on, when the PLC is not sending run commands to the VSD.

The PLC communication status indicates when the Ethernet communication to the PLC is healthy. The “Healthy” text is displayed and the green status indicator is on. The “Unhealthy” text is displayed and the red status indicator is on, when the Ethernet communication to the PLC has failed. The SCADA jog north and south option is selected by pressing the button. For jog north, the status indicator turns to green and the antenna starts moving in a north direction for 3

seconds. Once completed the status indicator turns to red and the antenna stops moving. During the on period the VSD runs the motor at 6Hz. The jog option is only possible when the SCADA is in manual mode (SCADA start/stop is stopped and the status indicator is red). The jog south option is similar to the jog north.

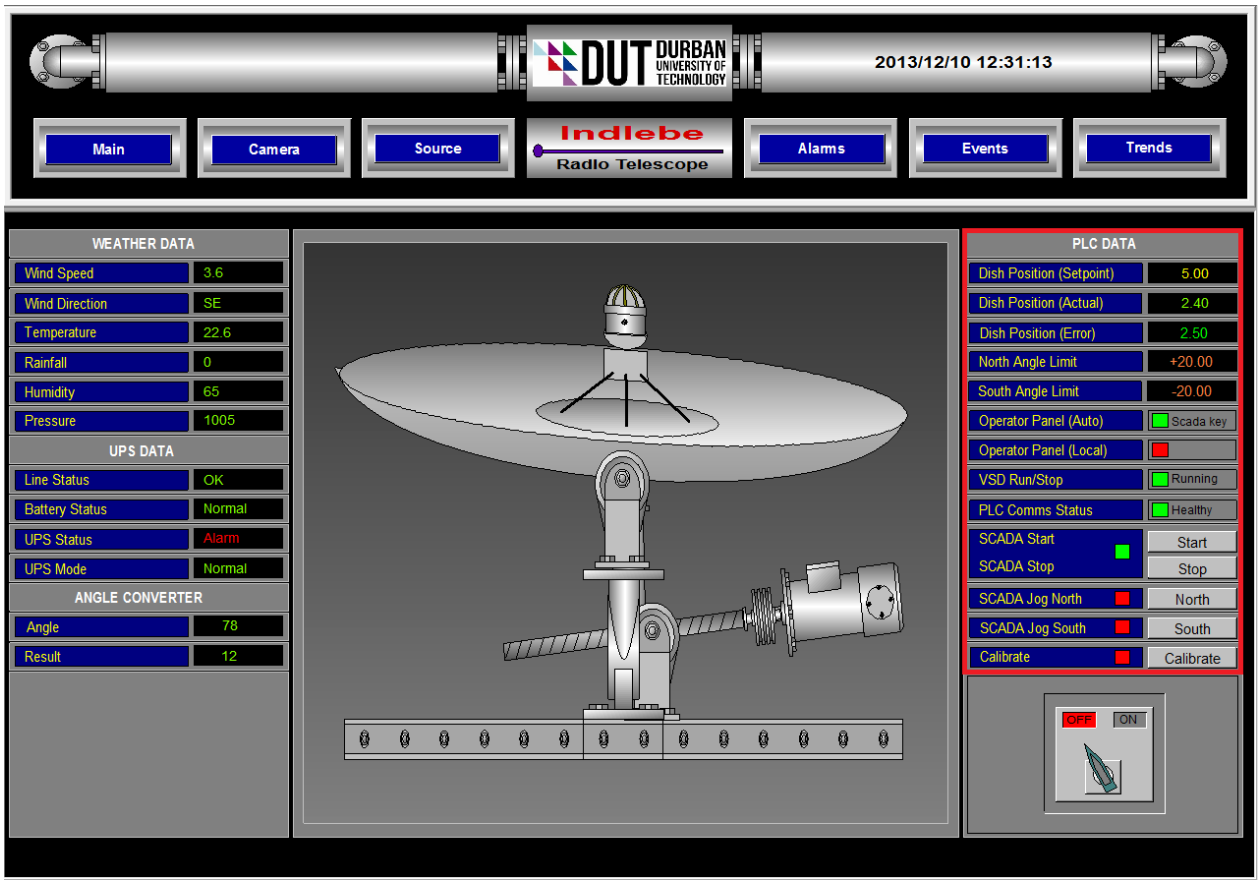


Figure 3.3 – PLC data on SCADA

The calibration option is used to calibrate the antenna position at the IRT 0° angle, which is equivalent to 90° elevation pointing perpendicularly to the Earth. Calibration is achieved by inputting a setpoint angle of 0°. If the antenna is not level then the jog function is used to get the antenna to a level position.

A digital level gauge is used to ensure the antenna is in the level position. The calibrate button is activated on the SCADA main overview screen to initialize the calibration function. The status indicator turns to green during the calibration process and then to red on completion. The PLC program saves the encoder value at the calibrate event and updates the “perpendicular to Earth” reference. The system also recalculates the maximum north and south limit values from this position.

The PLC data as displayed on the main overview screen of the SCADA system can be referenced as per **Figure 3.3** (as shown in the yellow box). The PLC system setup and configuration can be referenced as per **Appendix F**.

3.3 Weather station

3.3.1 Introduction

A weather station is a system that uses instruments and equipment to measure atmospheric conditions and provides information for prediction, analysis or control. These weather stations can be used by an individual at their residence or by an organization, solely intended for monitoring and distributing the data for analysis or control.

The weather station supplied by “Oregon Scientific” has been installed at the DUT. The purpose of this weather station is to provide real time weather data to the SCADA system for monitoring and trending. The measurements taken using the Indlebe weather station include; local

atmospheric pressure, rainfall, temperature, humidity, wind speed and direction. The details of this weather station can be referenced as per the manual [14].

3.3.2 Weather station communication

The instruments used, communicate wirelessly to a digital console that provides readouts of the data being collected. A block diagram of the system showing the communication protocols is shown in **Figure 3.4**.

The console interfaces to the Indlebe SCADA computer where the data is transferred every 1 second and displayed on the main mimic under weather data (refer to **Figure 3.5,** as shown in the red box) using scripts that update every 5 seconds. These updates are more than adequate for this operation.

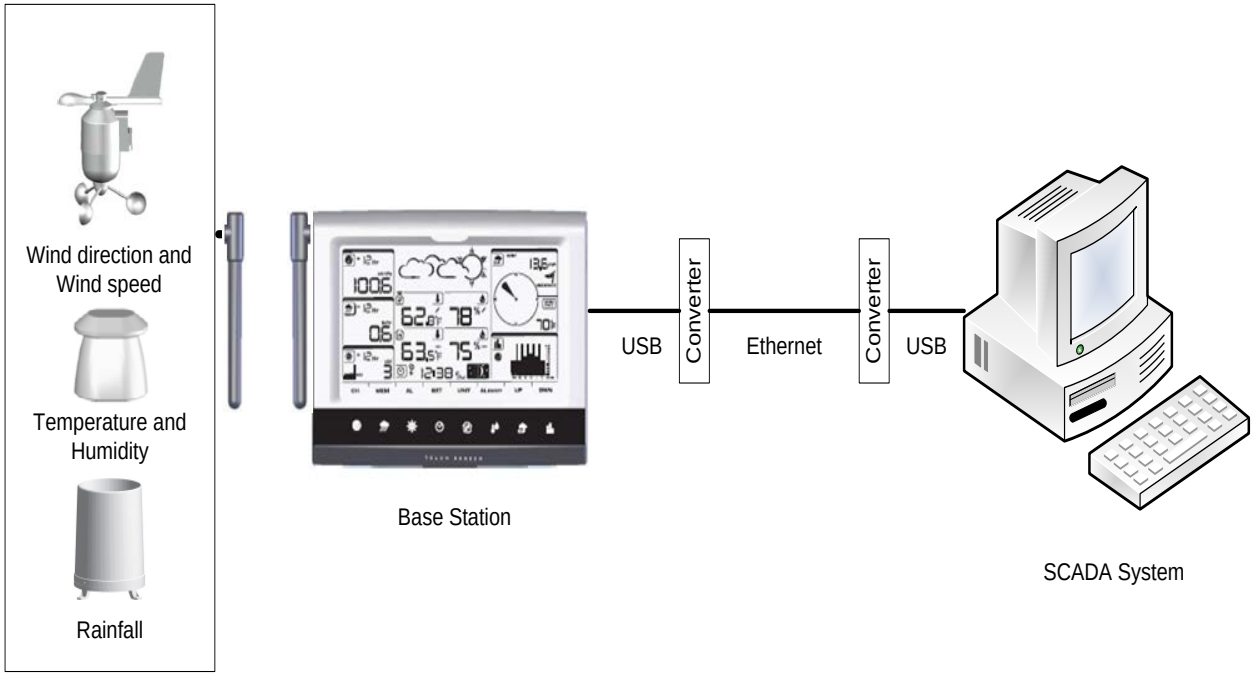


Figure 3.4 – Weather station communication block diagram

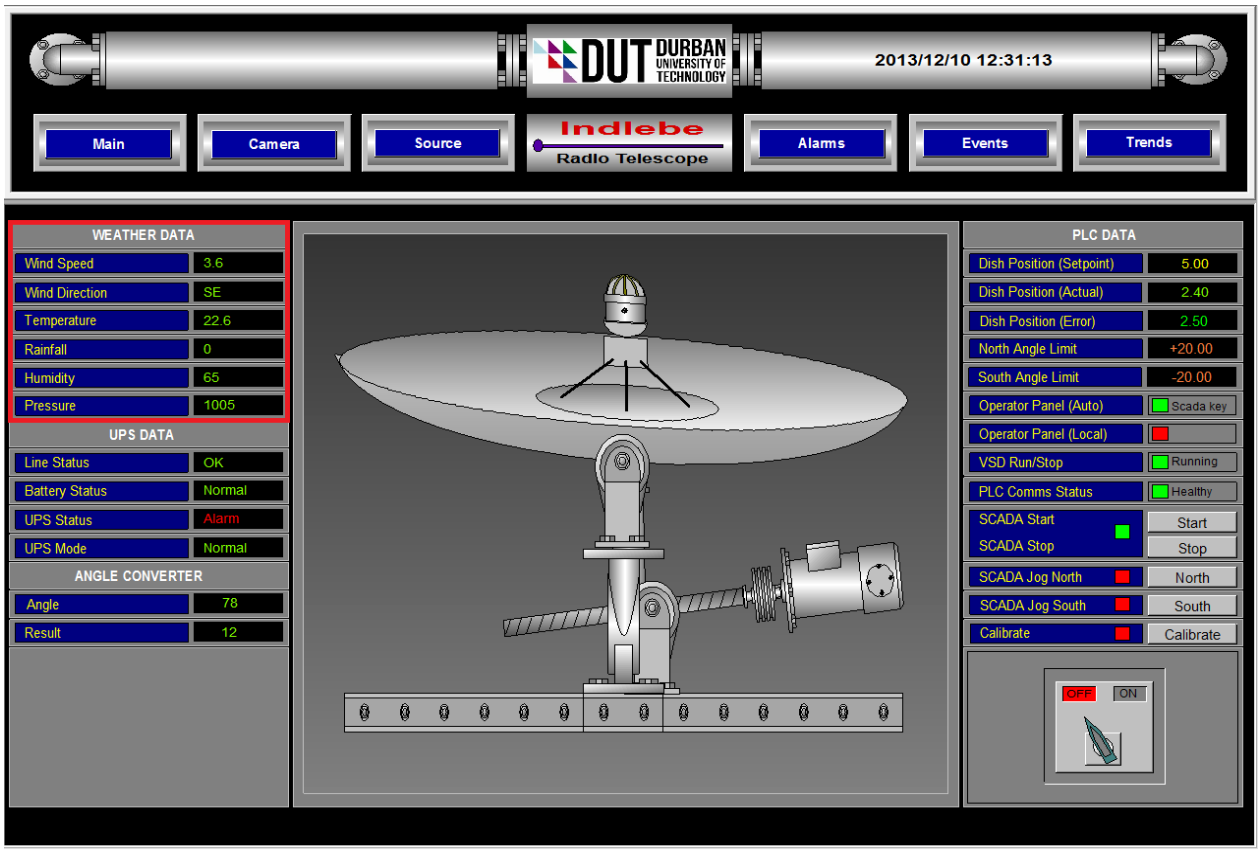


Figure 3.5 – Weather data information

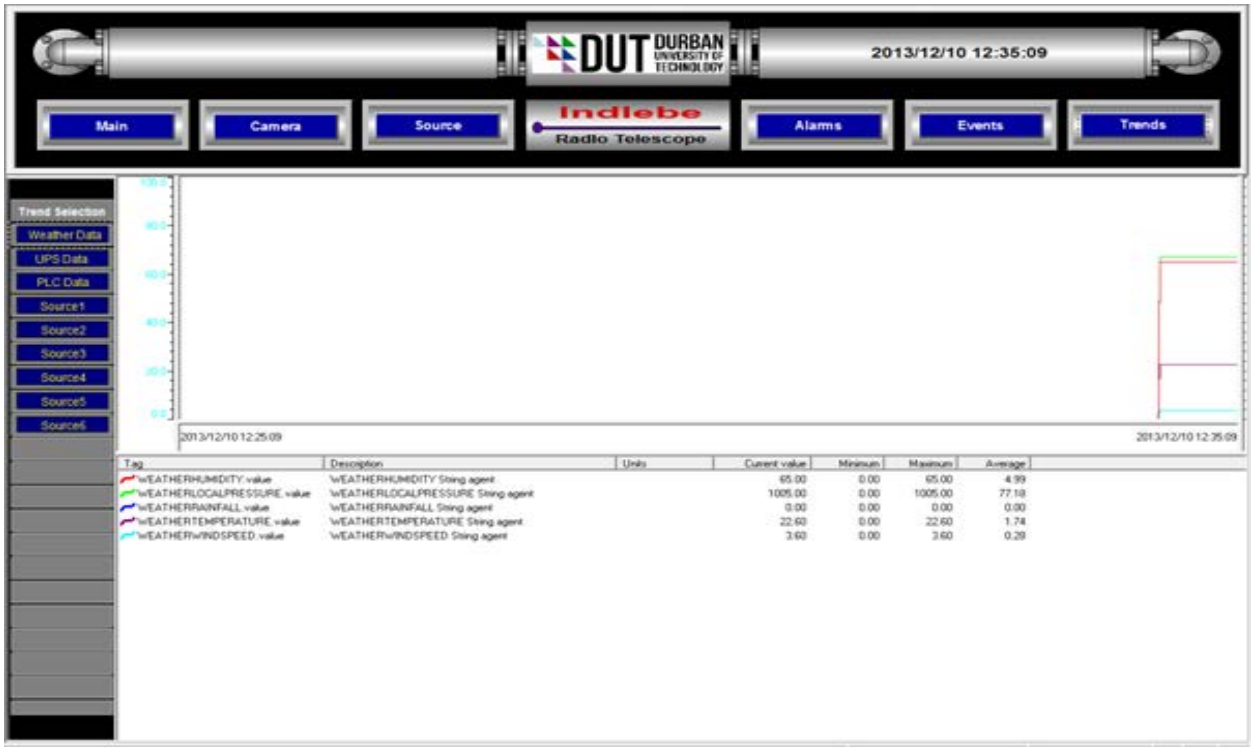


Figure 3.6 – Weather data trend

The weather data is also logged and trended as per **Figure 3.6**. The installation and configuration of the weather station can be referenced as per **Appendix G**. The scripts and its functionality can be referenced as per **Appendix H**.

3.4 Uninterruptible power supply

3.4.1 Introduction

A UPS is an electrical unit that is used to protect critical electrical loads (i.e. computer data centers, telecommunication systems or other electrical equipment) from sudden loss of mains power. A stable and reliable emergency backup power is provided by the UPS for a limited duration that is long enough to keep the required loads operational until a controlled shutdown is initiated. Data loss, data corruption and hardware damage due to electrical surges that impact the functioning of vital equipment, is prevented.

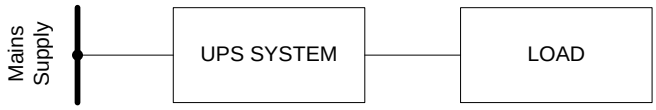


Figure 3.7 – UPS connection

The UPS system is connected between the mains supply and critical loads as illustrated in **Figure 3.7**. A UPS can be connected to a standard single phase utility input mains supply such as a three-pin 220VAC/16A socket and produce a single phase output or it can be connected to a three phase input supply and produce a three phase output.

To further improve the reliability of the system during a power failure, a standby generator may be incorporated into the system design to provide an alternate source of UPS input power. Such a generator must be able to start automatically and be sufficiently large enough to support the UPS on full load with a stable output supply.

3.4.2 Principle of operation

A static UPS system as illustrated in **Figure 3.8** includes a battery charger, power inverter, static switch and batteries. The batteries store energy and are charged when the mains supply is available. When there is a mains failure, the UPS transfers the load to the batteries. The battery charge is converted by the inverter into an alternating current (AC) supply to the load.

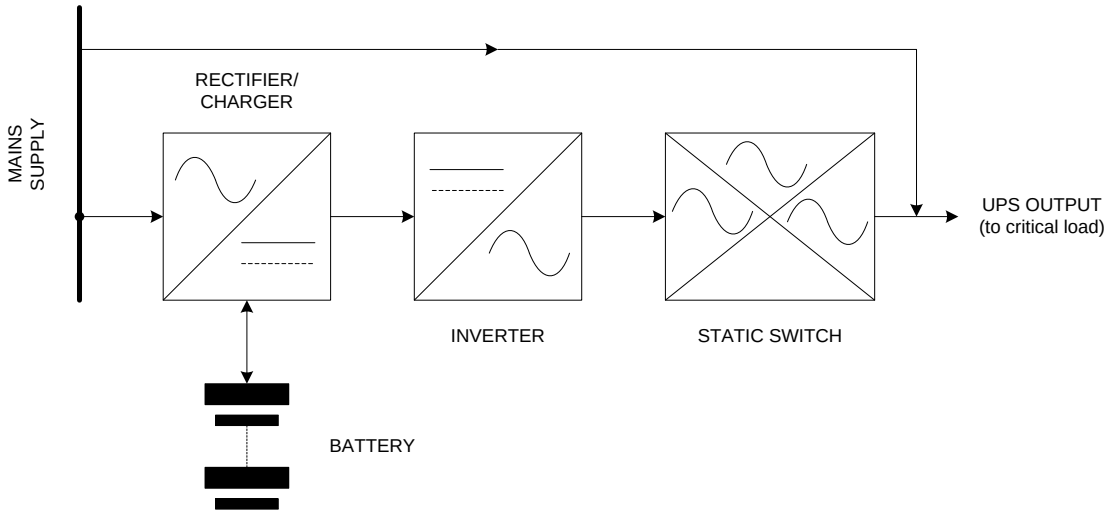


Figure 3.8 – Static UPS system

The batteries provide a power source for the inverter when the mains supply fails, whereupon it discharges at a rate determined by the critical load connected to the UPS output. The duration for

which the critical load can be supported in times of a mains failure depends upon the battery capacity and the percentage of applied load.

The UPS should contain sufficient battery capacity to support its fully rated output load for a specified time. When the battery capacity to the inverter falls below 25%, the inverter will automatically shutdown, thus switching off power to the load.

3.4.3 Background

Power outages and load shedding cannot be predicted as this is dependant on the demands by industrial, commercial and residential customers, has resulted in many organizations prioritizing this as its number one risk. DUT has installed a generator as its control measure to mitigate this risk, however at times during power failures the standby generator has not started up. The standby generator is located at S7 and only supplies that area of the S blocks. The radio astronomy group also decided to install a UPS specifically for the IRT data capture.

An EATON POWERWARE (PW9130i1500T-XL, 1500VA/1350W, 230V/6.5A) UPS is used to protect the data capture computer and the receiver electronics in the Indlebe Control Room, in the event of a mains failure or disturbance. With this equipment connected to the UPS, the load percentage is at 3% with a load current of approximately 0,2A (calculated as 3% of 1350W). The details of this UPS can be referenced as per the manual [15].

Due to this, it was important that the alarm and status information from the UPS be monitored and logged using the SCADA system. For this to be achieved, as illustrated in **Figure 3.9**, the

UPS had to be interfaced with the PLC and then to the SCADA system. A detailed UPS communication interface diagram can be referenced as per **Appendix I**.

Figure 3.9 – UPS communication interface

3.4.4 Uninterruptible power supply communications

The UPS alarm and status information can be communicated using one of 3 options, namely; volt-free contacts, RS-232 and Ethernet options. The Ethernet and RS-232 options where not chosen because of the limitation in integrating the data from the third party software provided into the SCADA system.

Therefore for this project the volt-free option was chosen. Using the volt-free option meant that this could easily be connected to the PLC and the data transferred through the existing Ethernet link between the PLC and SCADA system. The UPS only has a single relay output so an additional relay interface card that provided for an additional four relay outputs was installed. The specifications and functionality of the four relay outputs can be referenced as per **Appendix J**.

The following conditions of the UPS will be monitored and communicated to the SCADA system through the PLC; mains supply to the UPS on or off, battery voltage of the UPS normal or low, UPS in an alarm condition or not, and the UPS operating in inverter or bypass mode.

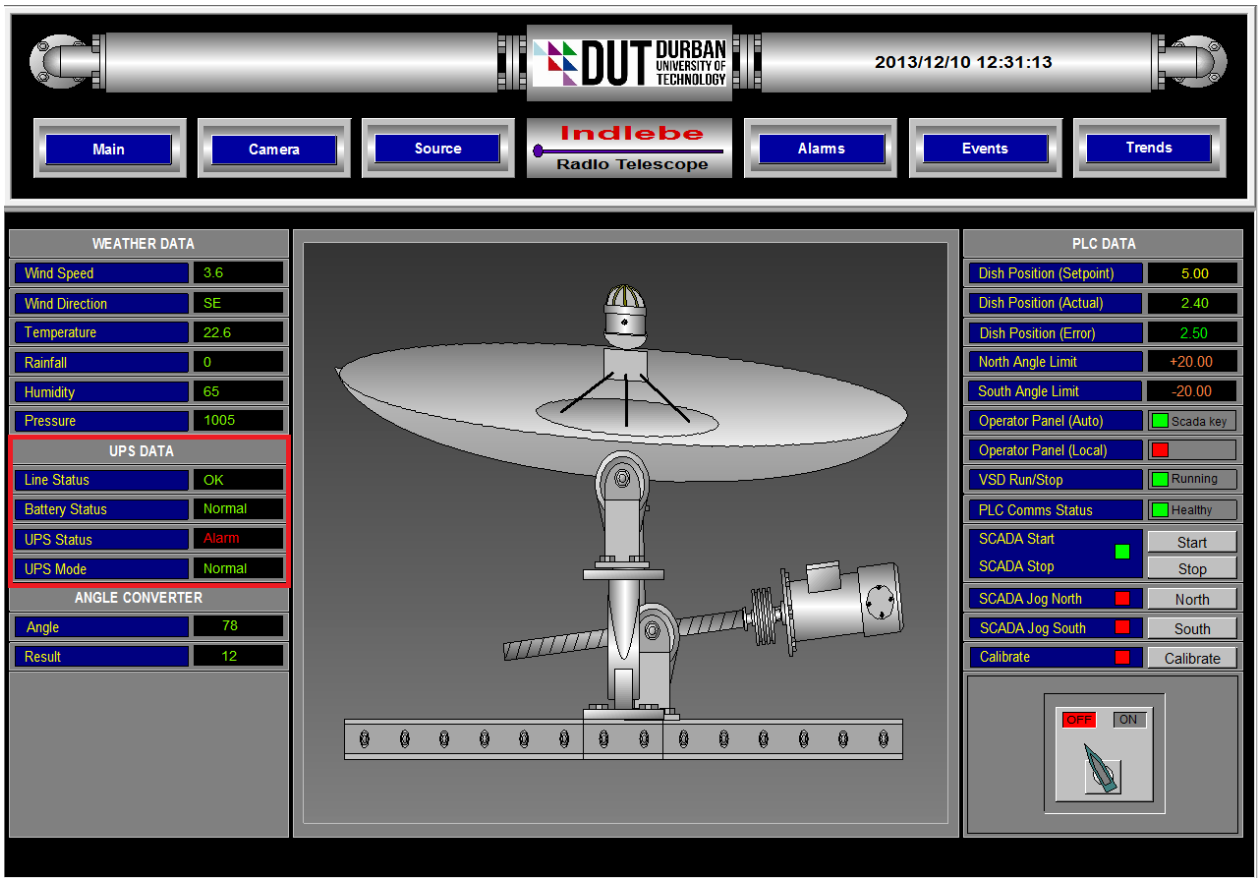


Figure 3.10 – UPS status information

These conditions are displayed on the SCADA system under UPS data and, are illustrated as per **Figure 3.10** (as shown in the red box). Displaying this information means that any fault with the UPS can be easily identified.

3.5 GSM Commander

3.5.1 Introduction

Data recording is fully automated and retrieved on a weekly basis due to the control room being locked and unattended for several days as a time. The control room has a data capture computer,

a SCADA computer, a network switch and the receiver electronics. The control room also has an air-conditioner that is used to maintain a stable room temperature as this affects the amplifier gains used to measure the required data and the life span of the UPS batteries. The rated capacity of a battery is based on an ambient operating temperature and any variations above the operating temperature alters the performance of the battery and shortens its expected life span.

In the event of a power loss, a UPS is used to keep the data capture computer and the receiver electronics operational for 2 hours. The backup time is an approximate time and varies with load, battery age and temperature. This signifies the importance of monitoring the power supply status and temperature of the control room, and informing relevant personnel of these events.

3.5.2 GSM Commander system

The GSM Commander is a device that is used for monitoring the power supply status and temperature of the ICR. Using a standard subscriber identification module (SIM) and antenna, it connects to the MTN cellular network for SMS communication. The monitoring functionality is configured by using a software program. The system has been configured to send an SMS only once when a condition is triggered and this requires no acknowledgement. The conditions that have been incorporated into the design include the following:

- General system status is communicated on the first day of every month and includes the date, time, signal strength, firmware version, serial number, airtime remaining and status of the mains power. The other reason for this condition is to ensure that the SIM-card is not blocked and remains active on the network.

- Mains power supply failure to the unit.
- Mains power supply restored to the unit.
- External battery voltage falls below 11,3 V.
- Measured temperature rises above 30 °C.
- GSM signal strength falls below 25 %.
- The prepaid airtime falls below 10 units.

The system is installed in two boxes with slide-on covers for easy accessibility. The boxes for the GSM Commander system can be referenced as per **Appendix K**. The first box houses the GSM Commander (GC0321) with general packet radio service (GPRS), a temperature interface module (GT002), a 0-100°C temperature probe (GT001-1) and a “pay as you go” SIM-card (071 715 7019).

The second box houses the 0.8Ah/12volt sealed lead acid battery (AS0.8-12) that is connected to the GSM Commander unit, as an external rechargeable backup battery. Also connected to the GSM Commander unit are an external antenna and a power supply unit (AC100-240V/14VDC, 500mA, PSU1405T). The system has been configured to send all messages to an administrator and an alternate contact person.

The hardware components and installation of the system can be referenced as per the GSM Commander general arrangement in **Appendix L**. Details on the GSM Commander can be referenced as per the manual [16].

3.6 Camera

3.6.1 Introduction

There are many types, designs, shapes and sizes of cameras available on the market. Each camera can be different in technology and application, however they are similar in purpose and functionality. Cameras can either be analog or digital and are used to capture or monitor images. These images (photo or recording) can be stored on a film/memory storage device or displayed on a visual device.

The IRT antenna is located outside the control room and visualization of the actual antenna position from the control room is limited, hence the need for a remote camera that would provide a real time image of the antenna. From a safety point of view, when the antenna is moving there is no way to determine if there is someone working on the antenna or not.

3.6.2 Camera system

A Sony SNC-CS10 indoor/outdoor industrial network camera was chosen. The details of this camera can be referenced as per the manual [7]. The camera is installed within an industrial housing and is mounted in a fixed position on the wall adjacent to the antenna, approximately three meters high. Using the local area network (LAN) port of the camera, it is connected to the same network switch as the SCADA system (refer to **Figure 3.11**). Having the SCADA and camera systems on the same network, the camera's Ethernet connectivity feature allows for the

image from the camera to be transmitted as data over the network connection, to the SCADA system to be displayed.

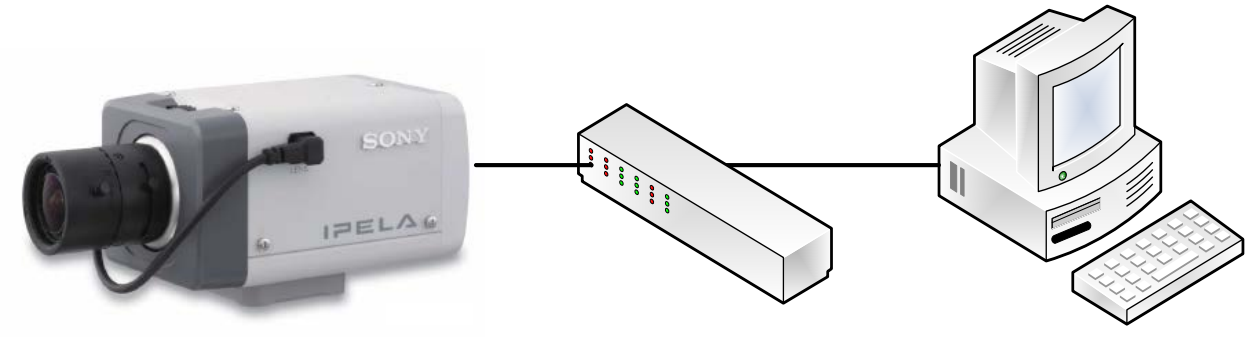


Figure 3.11 – Camera system

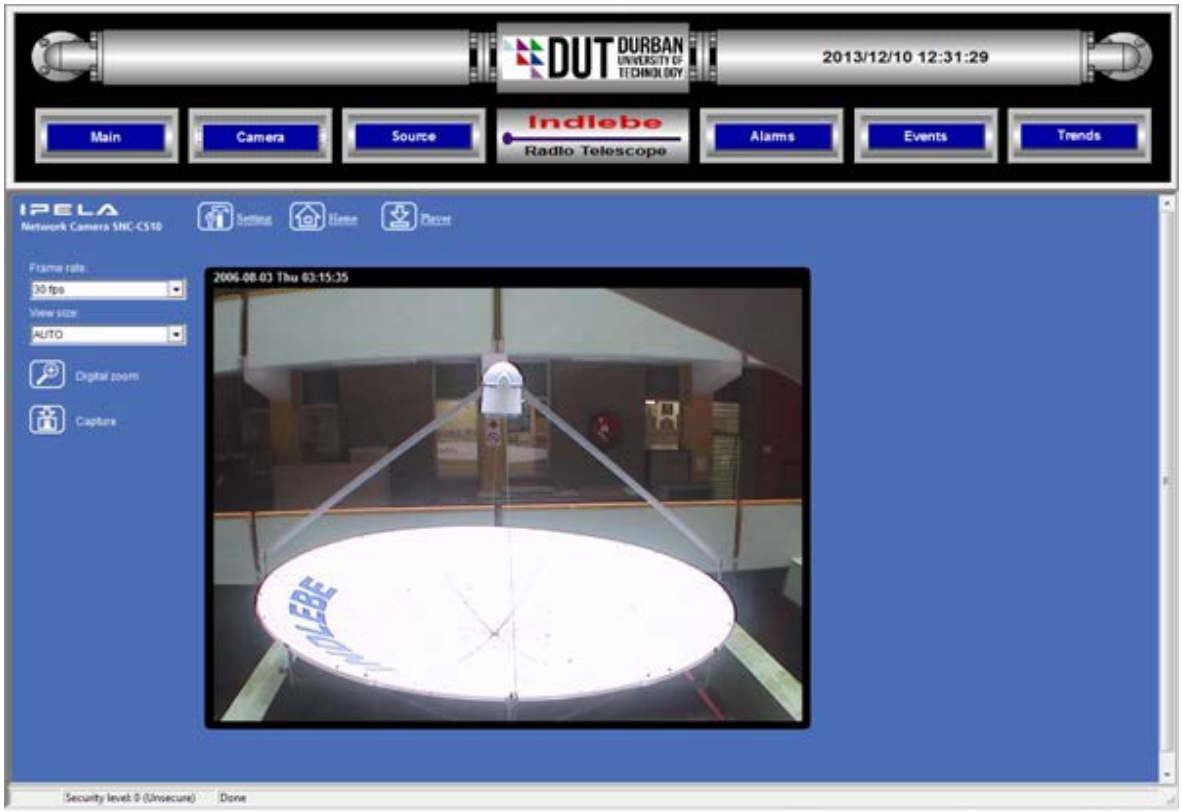


Figure 3.12 – SCADA camera display

The configuration and setup procedure of the camera and the SCADA system can be referenced as per **Appendix M**. By selecting the camera button on the header menu of the SCADA system, the real time image of the antenna is displayed (refer to **Figure 3.12**). The SCADA system is not used to store any images. The real time image from the camera gives a clear view of the antenna so that when the antenna is moved, it can be viewed remotely as it moves from one position to the next. The refresh rate of the camera is set to 30 frames per second.

3.7 Summary

The SCADA system has been designed with a main overview screen, providing all available information pertaining to the weather station, PLC and UPS. The navigation option allows for access to the camera, sources, alarms, events and trends. The interface between these sub-systems have been successfully implemented and tested. The control functionality that the SCADA system offers has also been tested successfully.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Introduction

The aim of this chapter is to present and discuss the results from the sub-systems and SCADA system using the data gathered. The challenges encountered are also discussed in this chapter.

4.2 Weather Station

The data for each weather condition is transferred from the weather station console to the SCADA computer and it is saved every one second to a cell location in a comma separated values (CSV) file for each weather condition. The SCADA system reads the data in the cell location using a script function for each weather condition and then displays the conditions on the main overview screen of the SCADA system. The weather station did function when it was installed, however it stopped working as a result of equipment failure and could only be tested using simulation methods. Testing the weather station system using simulation included changing the data value in the cell location of the CSV file for each weather condition. The new value was read successfully using the script function and displayed on the main overview of the SCADA system.

4.3 Camera

By selecting the camera tab on the SCADA header, it will navigate to the camera window where a view of the antenna is displayed. When the antenna is moved it can be viewed in real time as it

moves from one position to the next. Testing the camera system included observing the antenna movements during operation.

The antenna was moved from the -20° position to the 0° position and **Figure 4.1** shows the antenna in the 0° position. The camera system has the functionality of changing the view size of the image by clicking on the drop down option.

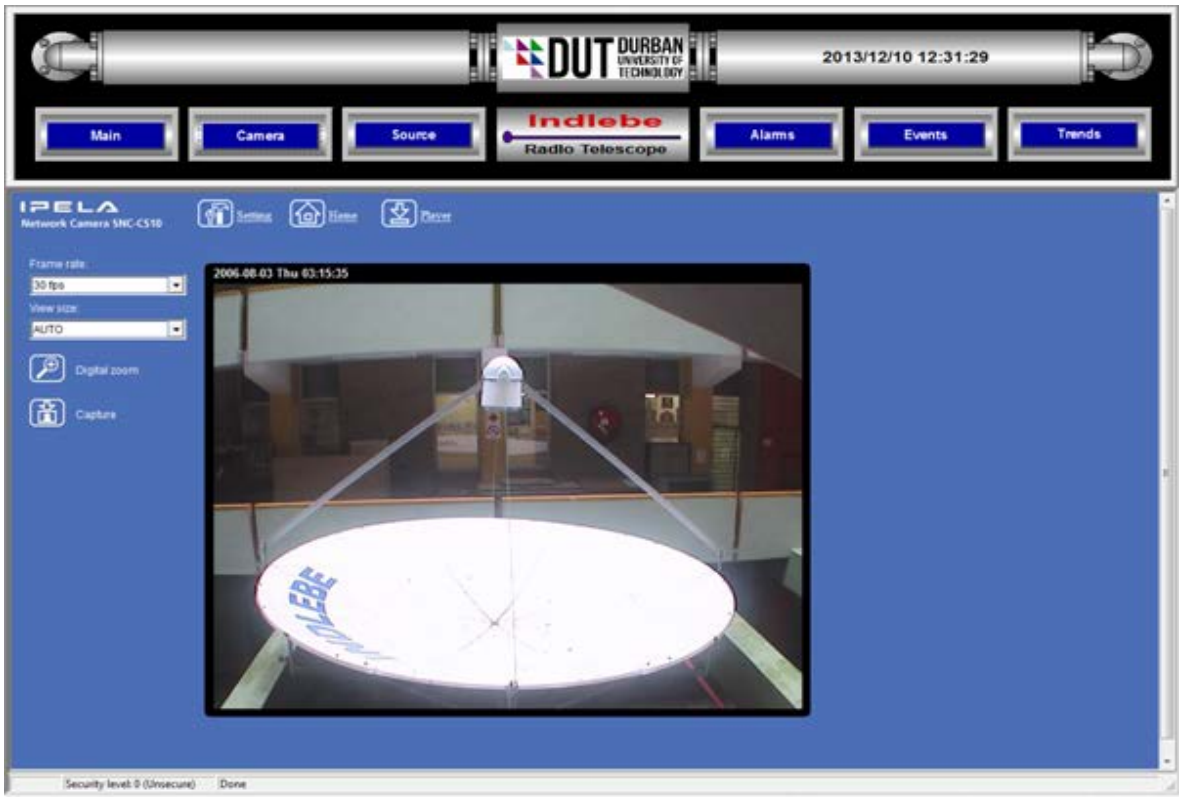


Figure 4.1 – Antenna at 0° position

4.4 Uninterruptible power supply

The UPS alarm and status information is communicated to the PLC using the volt-free contacts of the relay interface card of the UPS and the digital input card of the PLC. The information is

transferred through the Ethernet link between the PLC and the SCADA system and is displayed on the main overview screen of the SCADA.

Testing the UPS system was based on both specific test conditions and partial test simulation conditions. The results from the tests are shown in **Table 4.1**.

No	Test Conditions	UPS Alarm and Status Description	SCADA UPS Status Display
1	Switch on the mains power to the UPS	Line OK	Line Status OK
2	Switch off the mains power to the UPS	Line Failure	Line Status Failure
3	Connect pin 4 and 5 (15 pin D-type Plug)	Battery Normal	Battery Status Normal
4	Connect pin 4 and 6 (15 pin D-Type Plug)	Battery Low	Battery Status Low
5	Connect pin 7 and 8 (15 pin D-Type Plug)	UPS Alarm	UPS Status Alarm
6	Connect pin 7 and 9 (15 pin D-Type Plug)	UPS OK	UPS Status OK
7	Switch the UPS to normal mode	UPS Inverter On	UPS Mode Normal
8	Switch the UPS to bypass mode	UPS on Bypass	UPS Mode Bypass

Table 4.1 – UPS status and alarm simulation test results

4.5 GSM Commander

The GSM Commander has been configured to send an SMS when certain conditions are triggered. The test criteria and results for the conditions are as follows:

- The mains power supply failure condition was successfully tested by switching off the mains power supply to the unit. The result was that the system sent an SMS as "DUT Indlebe Control Room Power Failure".
- The mains power supply restored condition was successfully tested by switching on the mains power supply to the unit. The result was that the system sent an SMS that reads "DUT Indlebe Control Room Power Restored".
- A backup battery is connected to the unit as an additional source of power when the primary mains supply to the unit fails. The backup battery is mounted in a housing external to the unit. It is important to monitor the external battery voltage during a power failure as this determines when the unit will be switched off entirely. The external backup battery voltage falling below 11,3 volts was successfully tested by disconnecting the mains power and allowing the battery voltage to fall below 11,3 volts over a period of time. The result was that the system sent an SMS that reads "DUT Indlebe Control Room GSM Unit Backup Battery Voltage Below 11,3 Volts".
- The temperature rising above 30° C was successfully tested by holding the temperature sensor between two fingers and allowing the temperature to rise above 30 degree C. The result was that the system sent an SMS that reads "DUT Indlebe Control Room Temperature Above 30 Degrees".
- The signal strength below 25% was not tested, however this was successfully tested at a signal strength below 40% when installed. The result was that the system sent an SMS that reads "DUT Indlebe Control Room GSM Unit Signal Strength Below 40 Percent".
- The prepaid airtime below 10 units was successfully tested at a limit below 40 units. The result was that the system sent an SMS that reads "DUT Indlebe Control Room GSM Unit Prepaid Airtime Below 40 Units".

- The general system status was successfully tested. The result was that the system sent an SMS only on the first day of the month as “General Status: Date and Time: 06/08/2014-06:15:28, Signal Strength: 64%, Firmware Version: 7.21, Serial Number: 130321-30726, Airtime Remaining: 32 and Status of Mains Power: On”.

4.6 Manual data collection

To be able to successfully measure a celestial source depends on its position and time it will be passing overhead (because it is a transit telescope, it is actually the Earth that is rotating past the source). The time difference between the start time and the offset duration of the source will be the new start time the source will be passing over. The manual option allows the user to manually move the antenna to a desired position to measure a celestial source. The elevation angle of the desired source is determined and the user then enters the elevation angle as a setpoint into the SCADA system. The system displays this and moves the antenna from its current position to the new setpoint position. The antenna will remain in this position and the desired source is measured. The antenna will remain in this position until the user changes it. The manual option is restricted to only one source being measured and is intended for use as a once off data collection from a specific source.

To test this option, Sagittarius-A was chosen as the source to be measured at an elevation angle of +1°. This angle was then entered into the system as the setpoint. The system moved the antenna to the new setpoint position. The antenna was left in this position for a 24 hour period and the data collected is shown in **Figure 4.2** as a total power plot. The traverse of Sagittarius-A across the telescope beam is highlighted in the red circle.

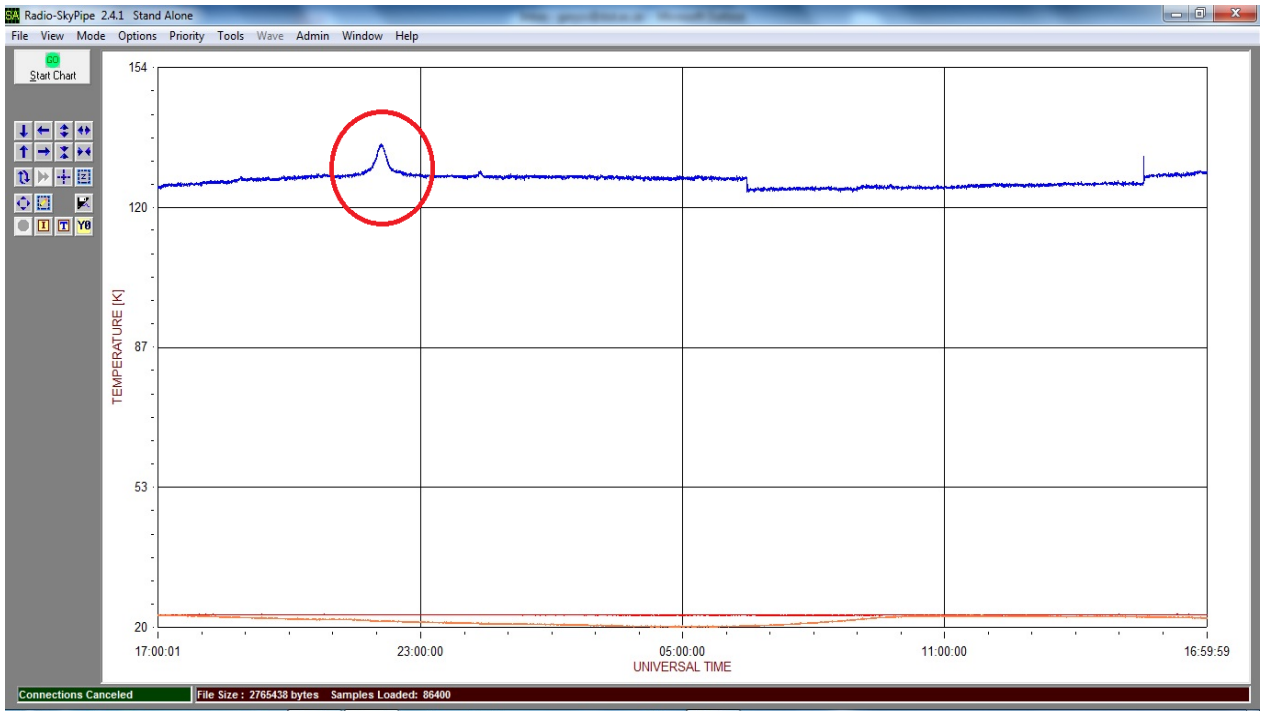


Figure 4.2 – Single source test results

The stable red line on the trend illustrates the temperature in the control room and the fluctuating orange line on the trend illustrates the temperature at the front end of the telescope. The time scale is based on universal time, that is, 2 hours behind local time.

4.7 Automatic data collection

Due to the requirement that the system is unmanned, an automatic functionality was required. In the manual mode the source peak would gradually drift off the recording, because the source would appear 3 minutes and 56 seconds earlier each day due to the movement of the Earth around the Sun. To correct this someone had to go to the control room and reset the time setting so that the peak moved back to the center of the trend. There are two options available for the automatic

functionality. The system has been designed where source one has been reserved to measure a single celestial source and sources two to six has been reserved to measure multiple sources.

The first option is to be able to measure a single source as determined by the start time, offset time, an elevation angle and a duration time. The ability to scan a large source over a period of several days or weeks, is to build up a picture of the source. The first option of the automatic control function allows the user to setup a single source for measurement from a maximum angle to a minimum angle with a 1° shift of the antenna position per 24 hours. Testing of this did not take place as it requires 41 days to complete a full measurement. The second option is to be able to measure multiple sources with each source having its own start time, offset time, elevation angle and a duration time.

The multiple source option was tested on the 14 June 2014 using the following sources:

- Vela-X with start time (universal) of 10:00:00, offset time of 00:03:56, elevation angle of -14° and the duration time of 00:00:00 (not used).
- Centauris-A with start time (universal) of 16:00:00, offset time of 00:03:56, elevation angle of -17° and the duration time of 00:00:00 (not used).
- Sagittarius-A with start time (universal) of 20:00:00, offset duration of 00:03:56, elevation angle of +1° and the duration time of 00:00:00 (not used).

The test results for the multiple sources is shown in **Figure 4.3**. A noise spike seen on the front-end temperature signal (orange), is generated by the VSD each time the antenna is moved.

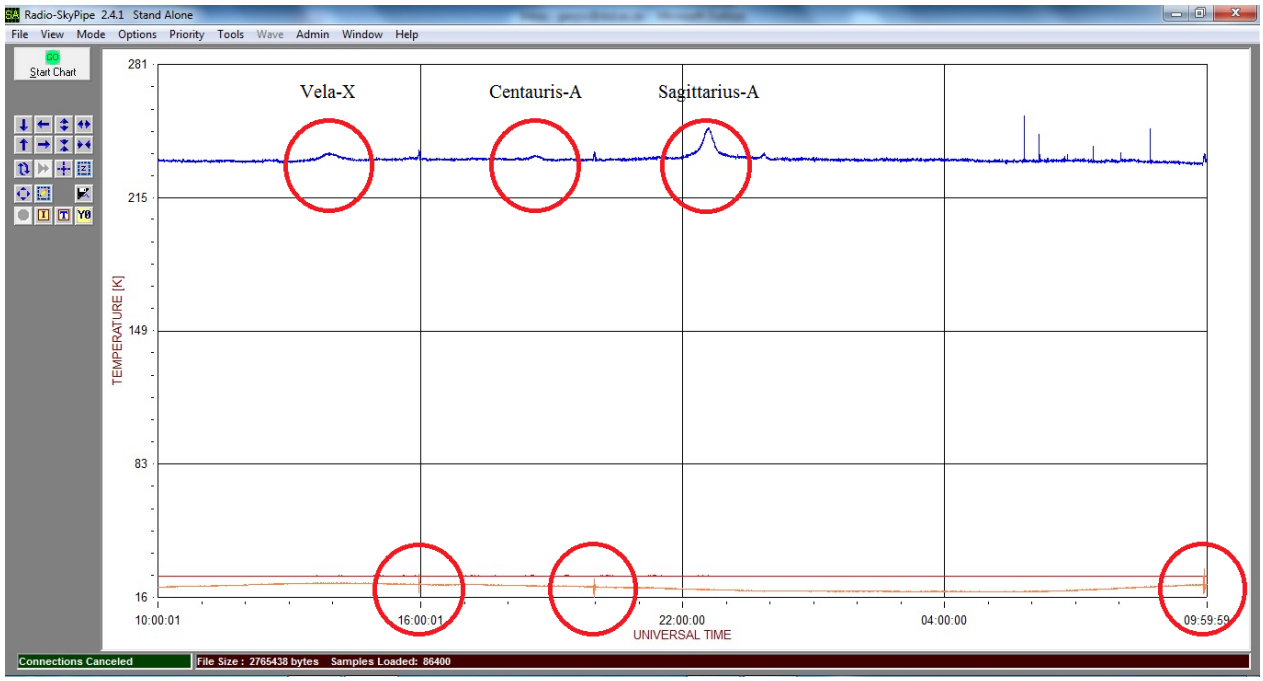


Figure 4.3 – Multiple source test results

The multiple source option allows for the antenna to be repositioned automatically for the different sources. Every day the system recalculates the new start times for each source and repositions the antenna automatically without any human intervention until the user stops the system. The major advantage of this is that there is no need for a user to be present at the control room to monitor or setup the system for each source. The data is collected for the duration of the measurement and can then be retrieved and analyzed later.

4.8 Challenges encountered

During the course of this project there were many challenges that were experienced. To be able to implement the Adroit SCADA system, detailed knowledge of the system had to be gained by studying technical information on Adroit [17, 18].

To be able to interface the SCADA system to and read/write data from the Omron PLC, knowledge of the PLC had to be gained. Given this complex technological environment and multitude of systems, this knowledge had to be leveraged to reduce the risk of implementation [19]. The weather station populated all its data to a CSV file. The challenge was to be able to read this data from the file and represent the information in the SCADA system. This was achieved using a visual basic script developed within the Adroit SCADA package. Integrating the camera view into the SCADA system is achieved by using the SCADA to call up a web path.

With no previous knowledge and experience in programming using “VB script”, proved to be very challenging in developing the control software.

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

The investigation of other observatories' control strategies led to consolidating all sub-systems into one control environment. The main objectives of the project were to integrate all the sub-systems into the SCADA system and to implement a control method that would move the antenna manually and automatically. The integration of the sub-systems namely; the weather station, the UPS, the GSM Commander, the camera and the PLC were successfully implemented. Once the integration was achieved, the next step was to control the antenna position manually and then automatically. Both control methods were successfully implemented and tested.

The previous control method of using the command line instruction to change the elevation angle meant that a user had to be available when measuring multiple sources. The biggest improvement this project achieved is the automatic control of the antenna position. Due to the ICR being unmanned a control method had to be designed to automatically reposition the antenna.

The measurement of a single source requires a user to input parameters into the SCADA system. These parameter changes always take place during normal working hours. The system will move the antenna to the parameter setpoints and measure the source.

Once the source passes over, the antenna remains in this position and the measurement continues even if there is no source. When the same source passes over again, it will be measured by the system as the antenna remained in the same position from its previous measurement. The antenna remains in this position until a new set of parameters are inputted by the user. The data collection

pc runs continuously storing the data on a 24 hour basis. The problem is to position the antenna so as to detect the relevant source. If multiple sources are to be measured and the manual option is used, then the first source parameters are inputted. Once the antenna has moved to the desired position and the source has been measured, then the user has to input the parameters of the second source. The parameterization may not always fall within the normal working hours of DUT, hence the user has to come in after hours to setup therefore the automatic control method was implemented with two setup options.

The first option is to setup multiple sources, to a maximum of five sources. The system is only able to measure three celestial sources outside our galaxy namely; Vela-X , Centauris-A and Sagittarius-A therefore a decision taken to incorporate five sources. Each source has its own parameter inputs which the user can setup during normal working hours. The user then selects the sources to be measured and enables the automatic function. The system will measure the selected sources and automatically move the antenna to the desired position as setup by the user. The same sources are then measured the next day (and for each subsequent day) but with a different start time as determined by the system using the offset time (3m 56sec). The antenna changes its angular position for each source based on the parameters set by the user and the priority of the source. When the user disables the automatic function, all the selected sources are disabled and the positioning of the antenna is stopped resulting in it remaining in its last position.

The second option of the automatic control function allows the user to setup a single source for measurement from a maximum angle to a minimum angle with a one degree shift of the antenna per 24 hours. Both automatic functions cannot be used simultaneously as their functionality differs. The automatic control functionality is easy to setup and the SCADA screen layout is

designed to be user friendly. The limits within the software protects the antenna from exceeding its maximum and minimum angular position and prevents the user from setting up the parameters incorrectly by displaying a warning message if this occurs.

The overall benefit of a SCADA system is that it offer features like a graphical interface system, alarm management system, communications sever and information historian, to assist operators in achieving a high level of efficiency [20]. In designing and implementing a SCADA system, it is important to display correct data to an operator so that they can easily identify which system or subsystem is at fault [21]. It is important to consider standardization when selecting the SCADA hardware, software, and network platform to use. By reducing variations in the system, training requirements and the use of external consultants can be significantly reduced [22].

To further enhance the system the following is recommended:

- Link the system to the DUT site network so that remote viewing can be achieved. The system would now become accessible to everyone that can logon to the DUT site network, users can get a better understanding of the system design, functionality and be able to monitor the data captured by the radio telescope. If the SCADA system is connected to the network or internet, proper system security measures must be implemented to protect the system against viruses, spyware and hackers [23].
- Install a separate UPS to supply power to the PLC and other computers in the ICR. The entire system would be protected and will remain functional during a power surge or power outage.

- To prevent the user from manually starting the software applications, the computers must startup automatically after a power failure.
- Include help options for the different functionality to assist a user in understanding the functionality of the different options available before selecting an option.
- Apply patches and updates to the operating system. Although the ICT is considered secure, the system password must be changed before the start of a new semester. This will protect the system from any unauthorized access of the system [24].

REFERENCES

[1] Rishin, P. Sabir, H and Aruna, J. 2005. *Adaptation of a Commercially Available Off-The-Shelf Multi Axis Motion Controller for Radio Telescope Control,* in *URSI Conference*. Delhi. 1-4.

[2] vd Schaaf, K. and Lawerman, E. 2005. *SCADA in the LOFAR Radio Telescope*, in *10th ICALEPCS International Conference on Accelerator and Large Expt. Physics Control Systems*. Geneva. 1-6.

[3] Daneels, A. and Salter, W. 1999. *What is SCADA*, in *International Conference on Accelerator and Large Experimental Physics Control Systems*. Italy. 339-343.

[4] Bailey, D. and Wright, E. 2003. *Practical SCADA for Industry*. Oxford: Elsevier.

[5] *Adroit Technologies sponsors DUT*. 2009. Available: http://instrumentation.co.za/news.aspx?pklnewsid=32825 (Accessed 08 July 2009).

[6] Shahzad, A. Musa, S. Aborujilah, A and Irfan, M. 2014. *The SCADA Review: System Components, Architecture, Protocols and Future Security Trends.* American Journal of Applied Sciences, **11**(8): 1418-1425.

[7] *Sony IPELA SNC-CS10/CS11 Network Camera*. 2007. Available: http://www.manualslib.com/manual/321509/Sony-Ipela-Snc-Cs10.html (Accessed 12

August 2014).

[8] Brooks, CJ. 1999. *A+ Certification.* 2nd ed. New York: Pearson.

[9] Wilson, M. 1999. *Connections.* Printed Circuit Design, **16**(3): 48-48.

[10] Computer Technology Industry Association. 2002. *Network + Certification Support Skills Volume 1*. London: GTS Learning International Limited.

[11] Schneider, DI. 1999. *Computer Programming Concepts and Visual Basic*. United States of America: Pearson Custom Publishing.

[12] Novoselov, V. 2011. *Use of encoders to measure the position of servomotors in the electrohydraulic systems for automatic control of steam turbines produced by the ZAO Ural Turbine Works.* Thermal Engineering, **58**(1): 26-27.

[13] Coulson, G. 2013. Radio Telescope PLC and SCADA Manual. Eastcoast Solutions cc, 2011 (Unpublished user manual).

[14] Oregon Scientific, Inc. 2012. *Professional Weather Center Model: WMR200/WMR200A User Manual*. Available: http://www.manualslib.com/download/116216/Oregon-Scientific-Wmr200a.html.

[15] *Eaton 9130 UPS 700-3000 VA User's Guides*. 2012. Available:

http://www.manualslib.com/manual/41745/Eaton-Electrical-Powerware-9130.html (Accessed 11 December 2013).

[16] *GSM Commander Manual*. 2011. Available: http://gsmcommander.com/download/ (Accessed 15 June 2011).

[17] *Technical Description Adroit Version 7*. 2010. Available: http://www.adroit.co.za/download/ (Accessed 11 June 2013).

[18] *Quick Start Guide - Getting Started With Adroit*. 2009. Available: http://www.adroit.co.za/download/ (Accessed 02 February 2009).

[19] Ketchledge, J. 2005. *Succeeding at Systems Integration.* Energy Markets, **10**(3): 24-28.

[20] Hanks, K. 1998. *SCADA offers enhanced reliability to product transmission industries.* Instrumentation & Control Systems, **71**(1): 43-47.

[21] Osborne, WH. 1987. *Choosing a SCADA system with the operator in mind.* IEEE Transactions on Industry Applications, **1A-23**(6): 1073-1076.

[22] Graham, N and Matthew, P. 2011. *SCADA standardization*. InTech, **58**(3): 18-20,22.

[23] Robles, RJ and Choi, M. 2009. *Assessment of the Vulnerabilities of SCADA, Control Systems and Critical Infrastructure Systems.* International Journal of Grid and

Distributing Computing, **2**(2): 27-34.

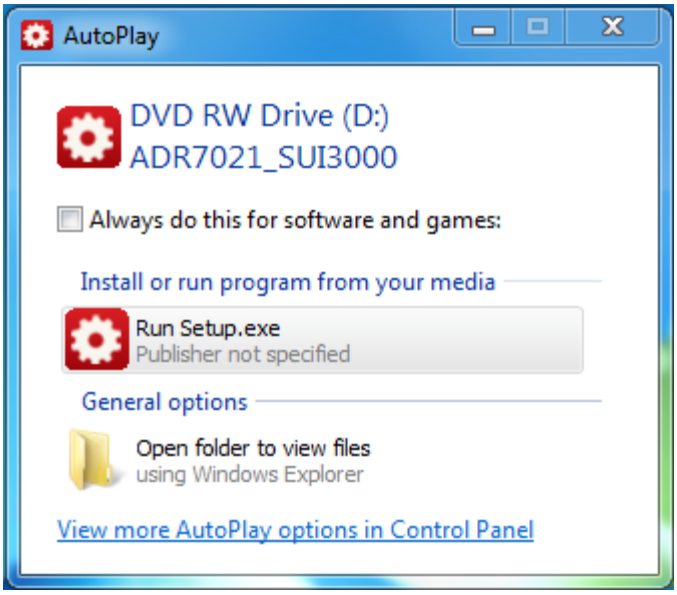
[24] Brown, T. 2005. *Security in SCADA systems: how to handle the growing menace to process automation.* Computing and Control Engineering Journal, **16**(3): 42-47.

APPENDIX A

SCADA SOFTWARE INSTALLATION PROCEDURE

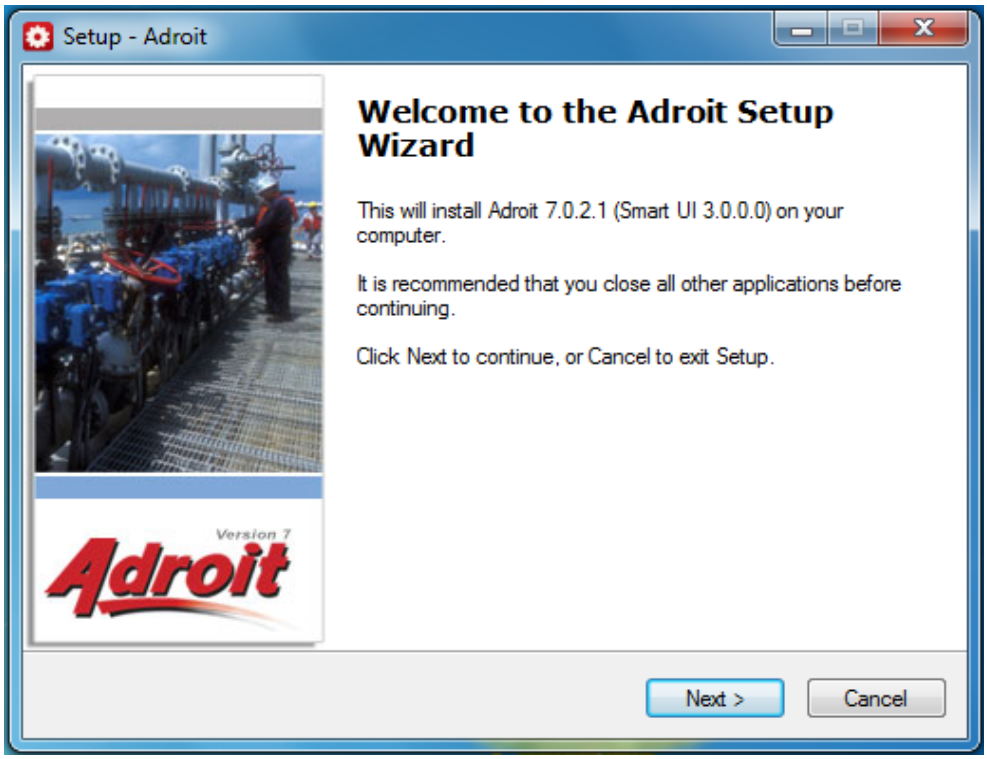
The SCADA software used for this project Adroit version 7. The procedure below is to be followed when installing the SCADA software:

- Click on “Run Setup.exe”.

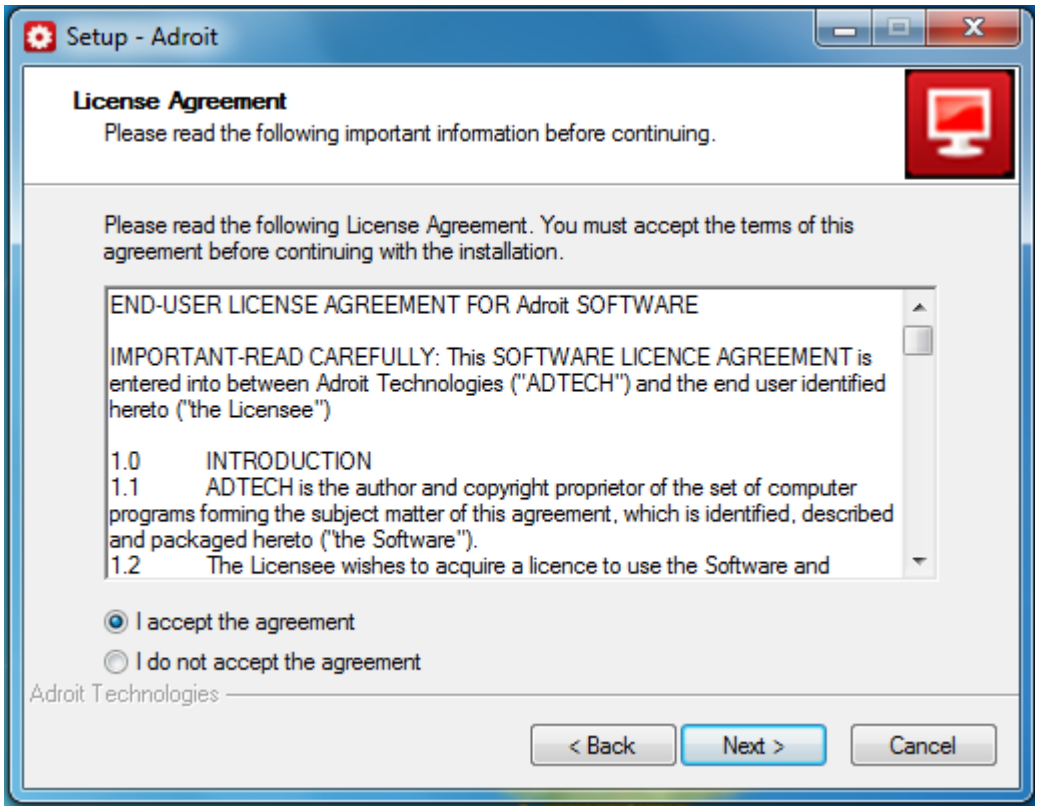


- Use account control – select “Yes”.
- At the Install screen, click on “Install”.

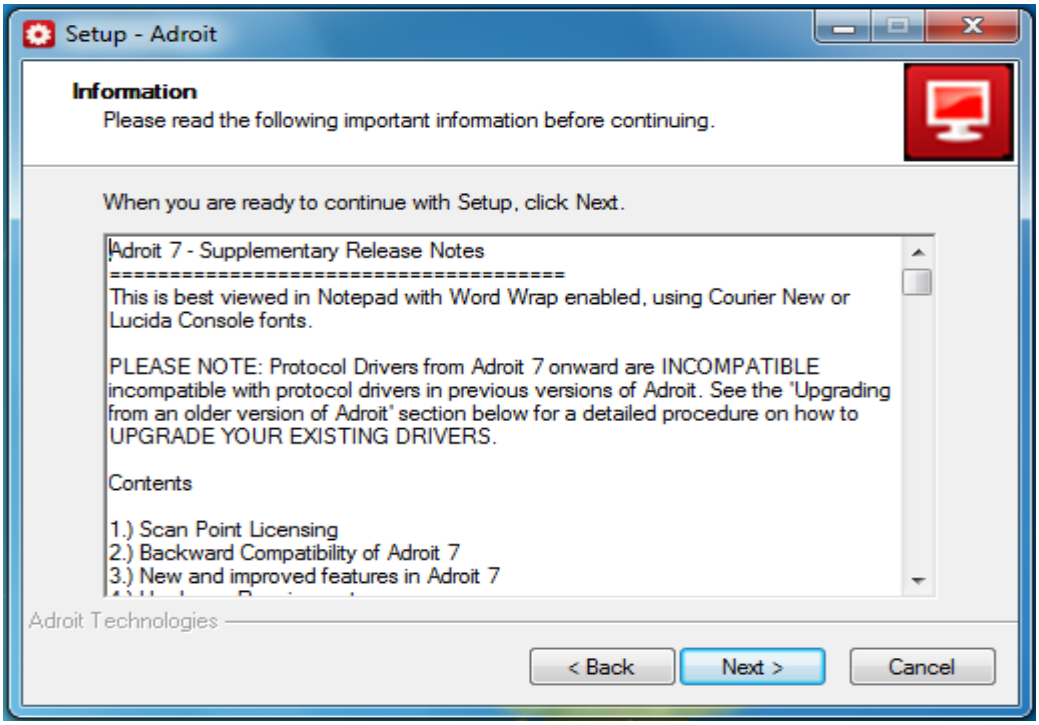
- Setup Adroit: Welcome Screen – click on next".



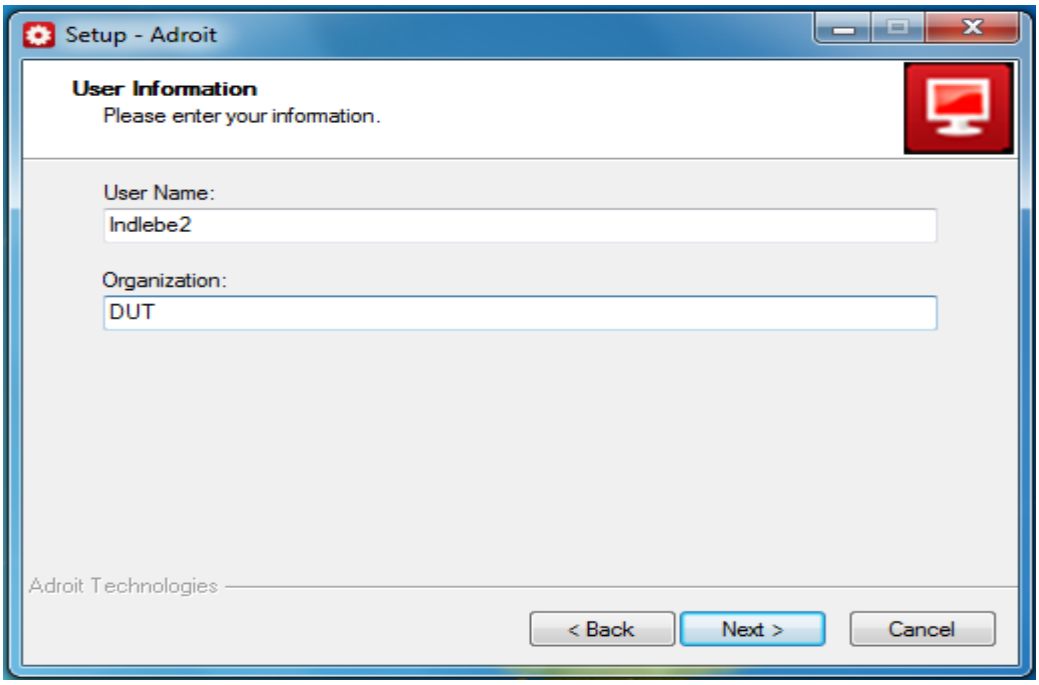
- Setup Adroit: License agreement screen – click on “I accept the agreement” and then click on “Next”.



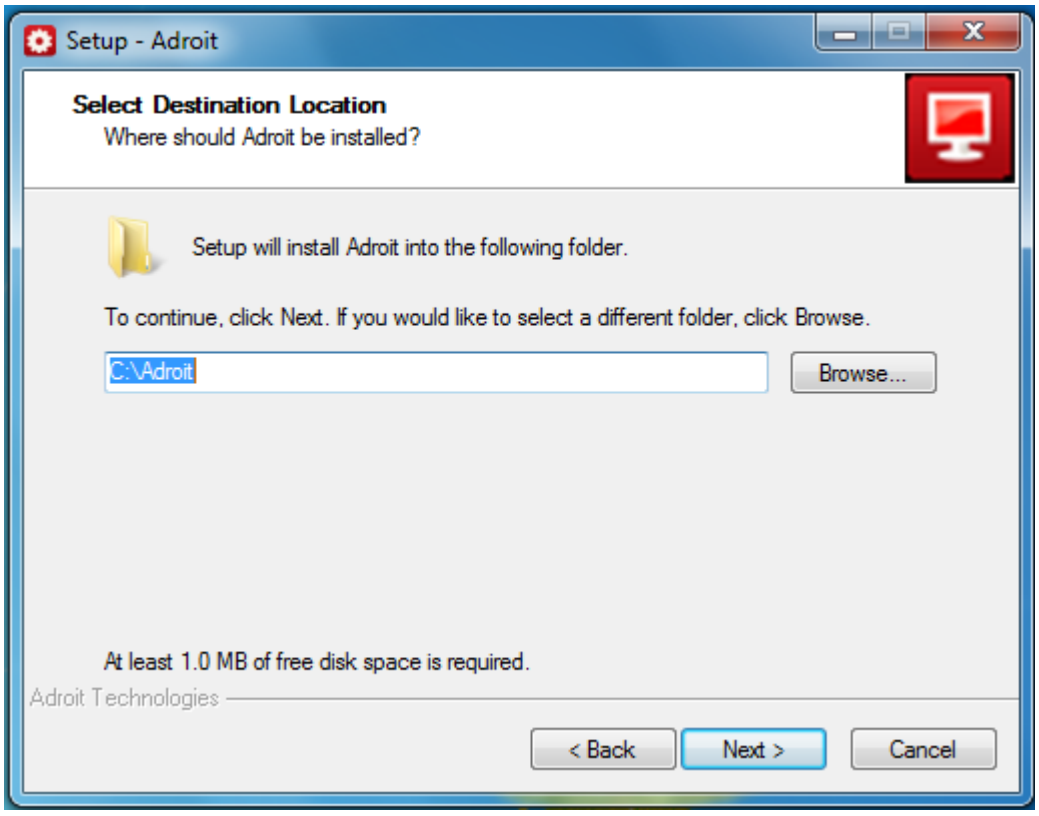
- Setup Adroit: Information screen – click on “Next”.



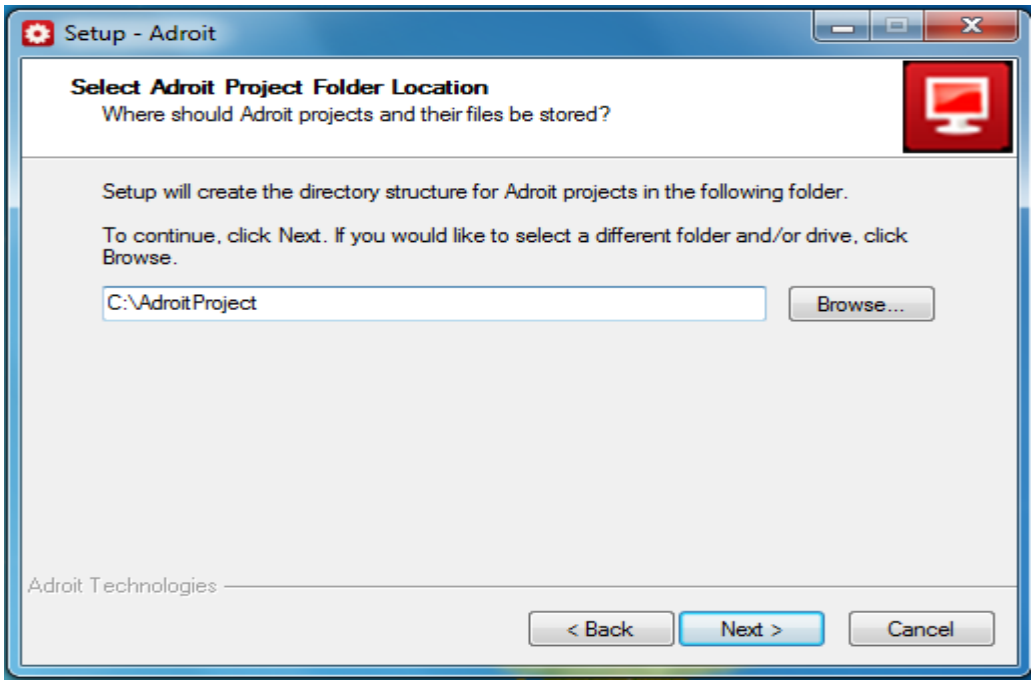
- Setup Adroit: User Information screen – User Name: Indlebe2, Organization: DUT, click on “Next”.



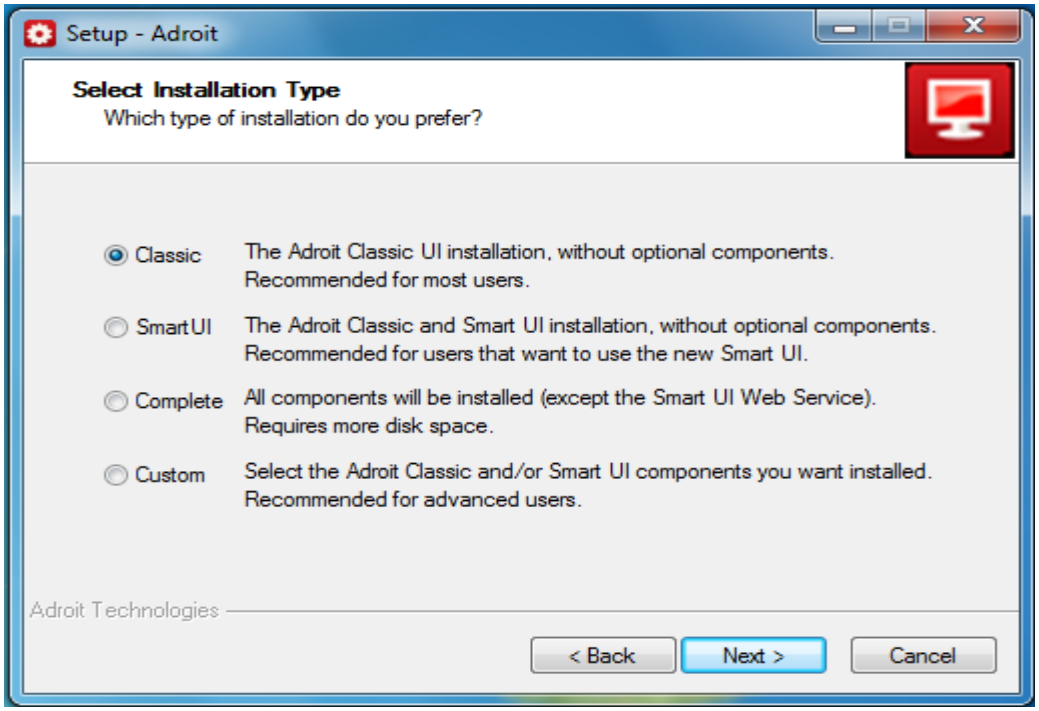
- Setup Adroit: Select Destination Location screen – select “c:\Adroit” and then click on “Next”.



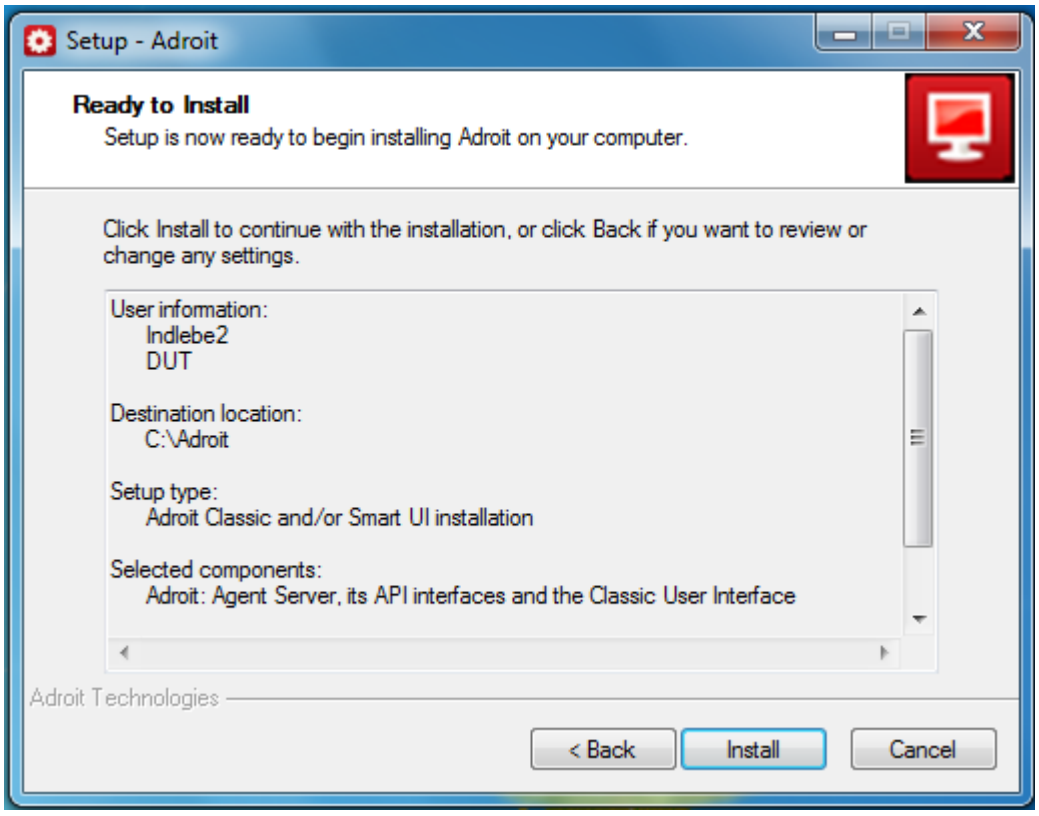
- Setup Adroit: Select Adroit Project Folder Location screen – select “c:\Adroit Project” and then click on “Next”.



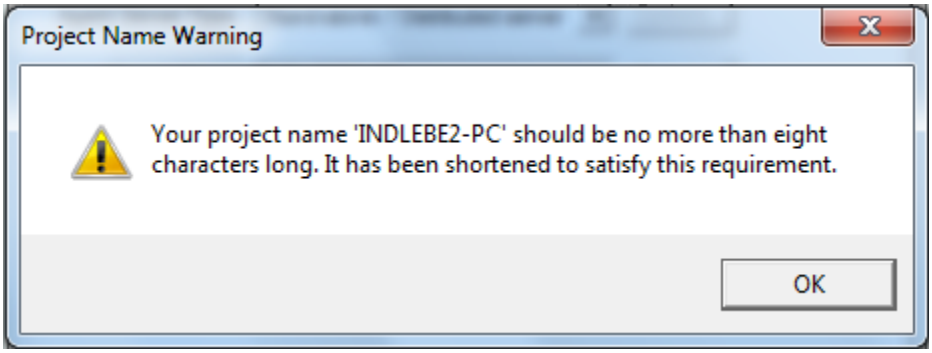
- Setup Adroit: Select Installation Type screen – click on “Classic” and then click on “Next”.



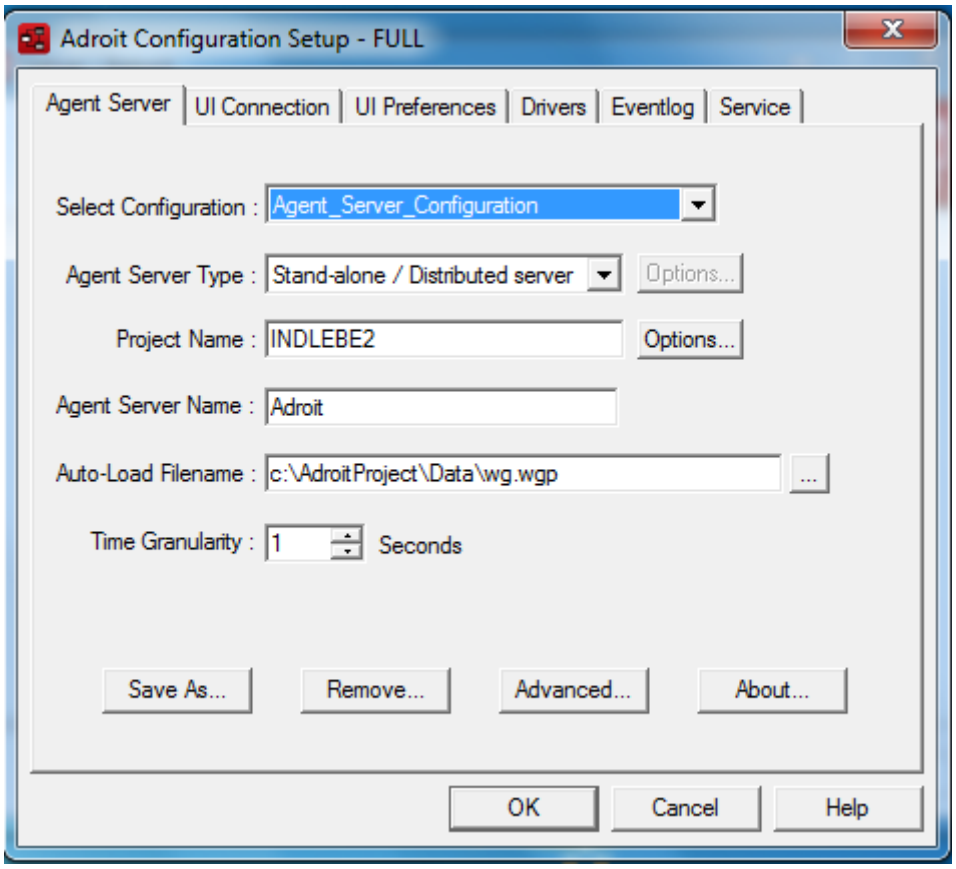
- Setup Adroit: Ready to Install screen – click on “Install”.



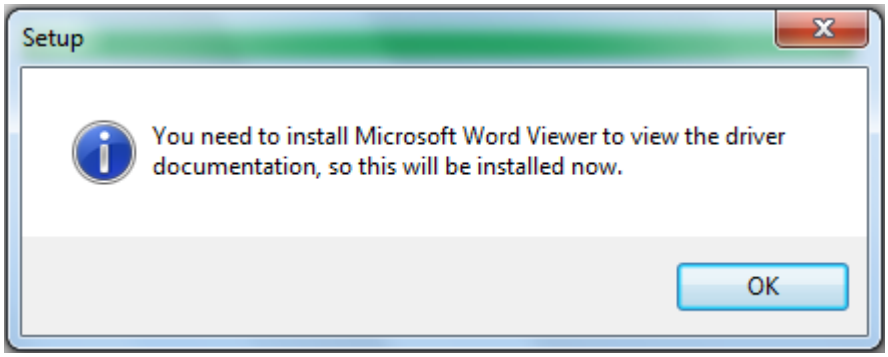
- Project Name Warning screen – click on “OK”.



- Adroit Configuration Setup – Full screen – click on “OK”.

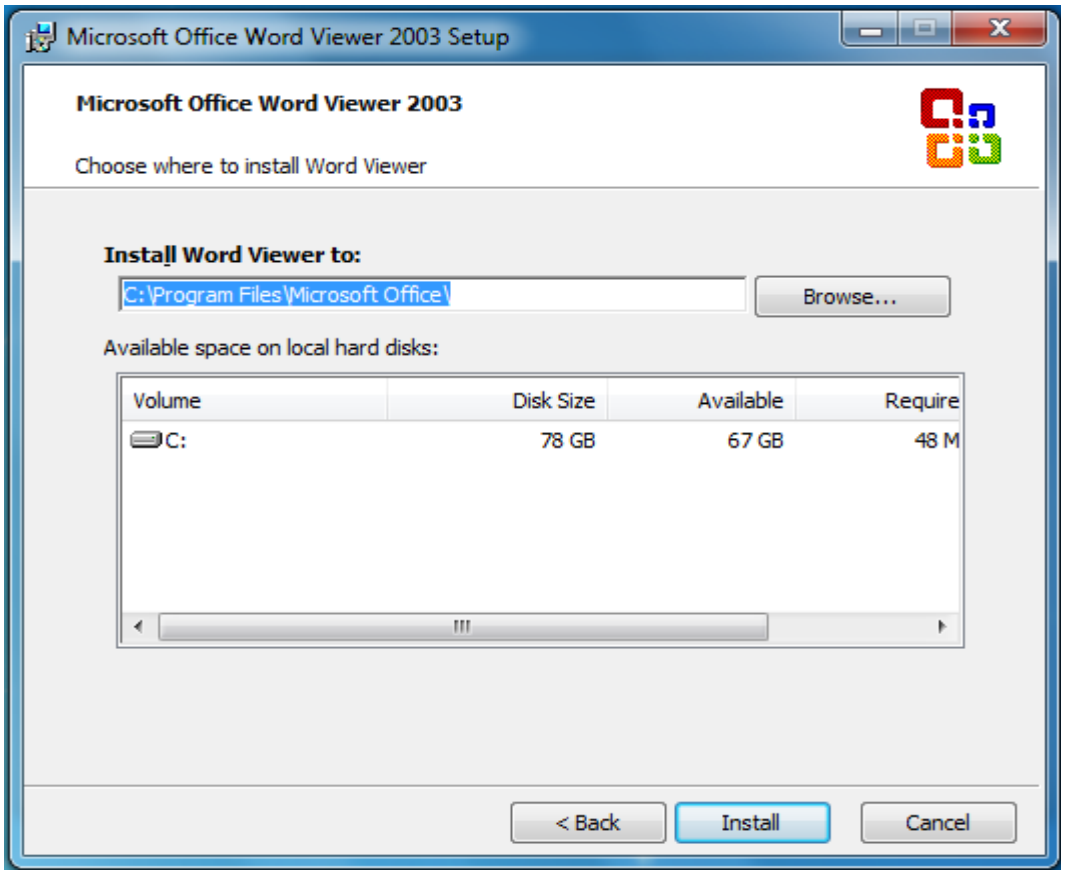


- Setup screen – click on “OK”.

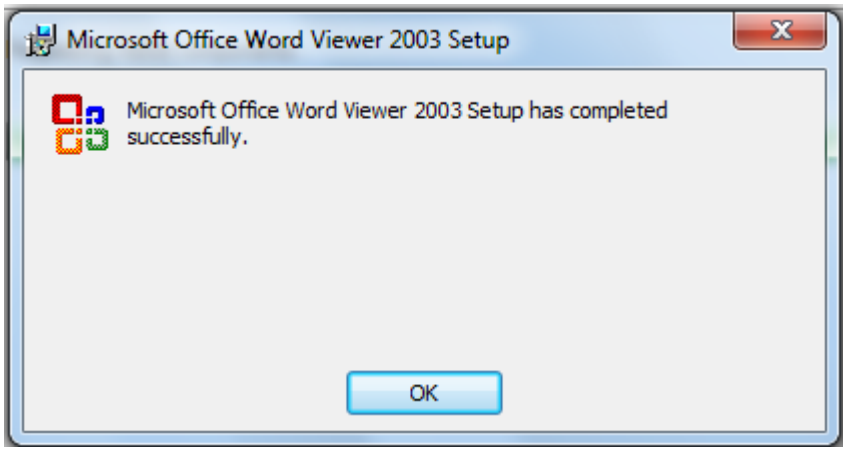


- Microsoft Office Word Viewer 2003 Setup: License Agreement screen – select “I accept the terms in the license agreement” and then click on “Next”.

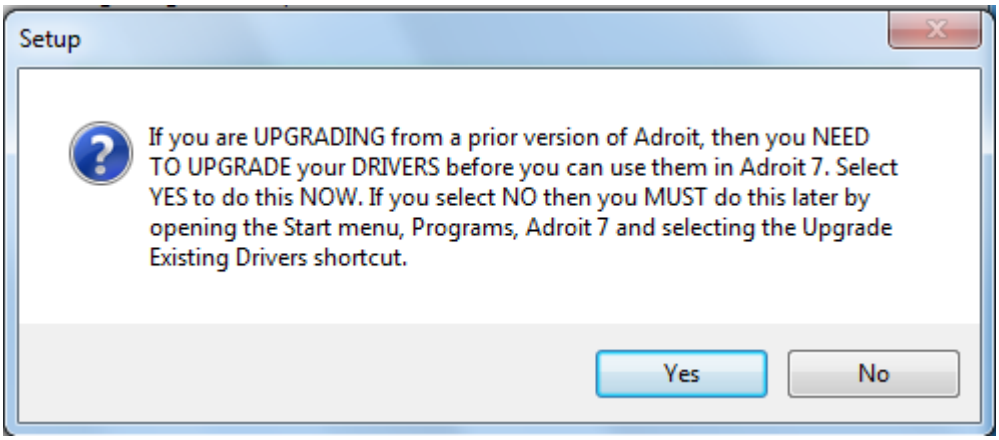
- Microsoft Office Word Viewer 2003 Setup: Installation screen – click on “Install”.



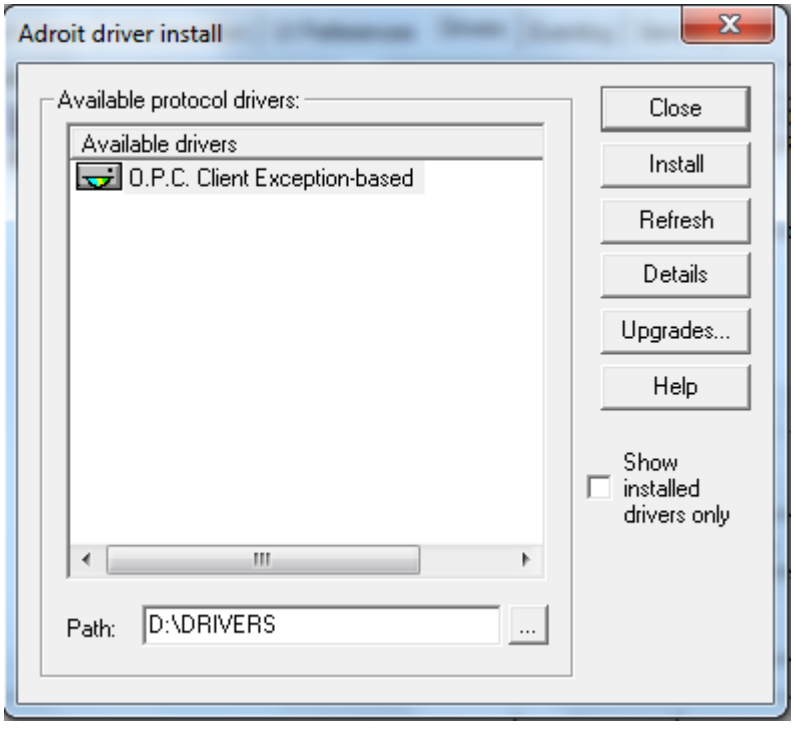
- Microsoft Office Word 2003 Setup screen – click on “OK”.



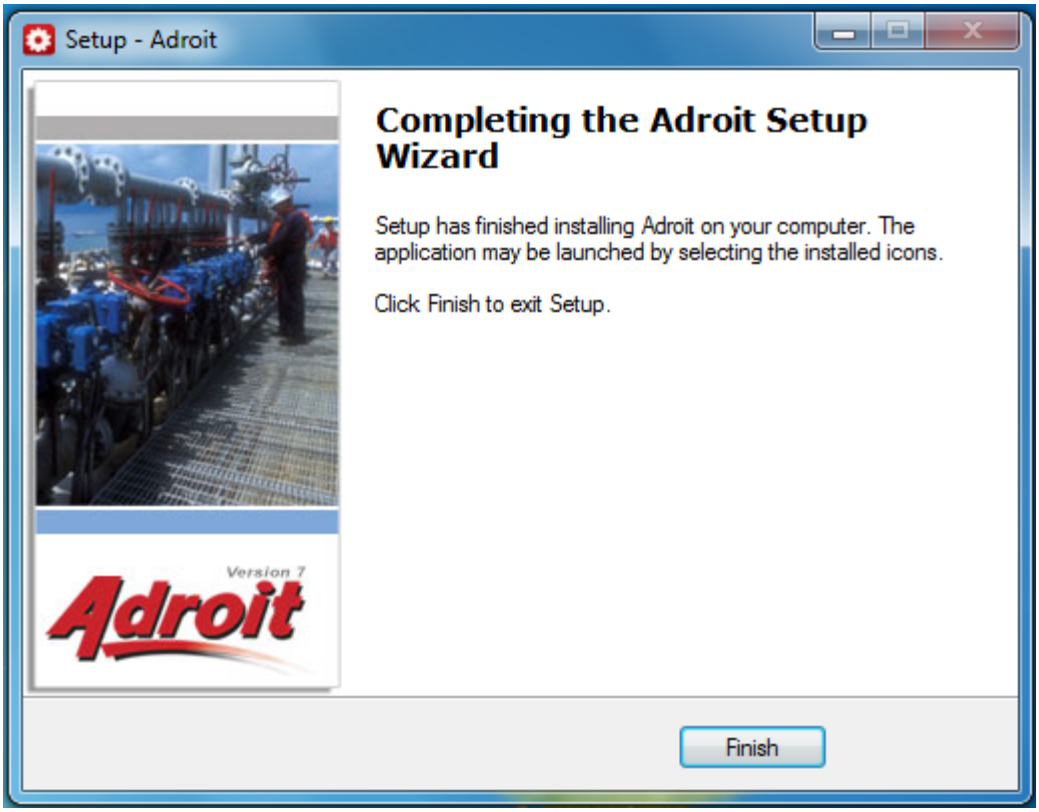
- Setup screen – click on “Yes”.



- Adroit Driver Install screen – click on “Close”.



- Setup Adroit: Completing the Adroit Setup Wizard screen – click on “Finish”.



- Install the Adroit Dongle into one of the USB ports (Adroit Dongle – serial number: 8780, scan points: 750 and remote nodes: 2).
- Device Driver Software Popup screen – click here for details.
- Drive Software Installation screen – click on “Change Settings”.
- Device Installation Settings screen – select “Yes, do this automatically (recommended)”.

NB: when installing the Adroit software (version 7), if the following errors occur;

- OMRON, not licensed to use Adroit Protocol Drivers.
- Program compatibility assistant, this program might not have installed correctly, program: unknown publisher, location: d:\setup.exe.
- Device driver software was not successfully installed, HASP HL3.21, no driver found.

Then do the following to prevent these errors:

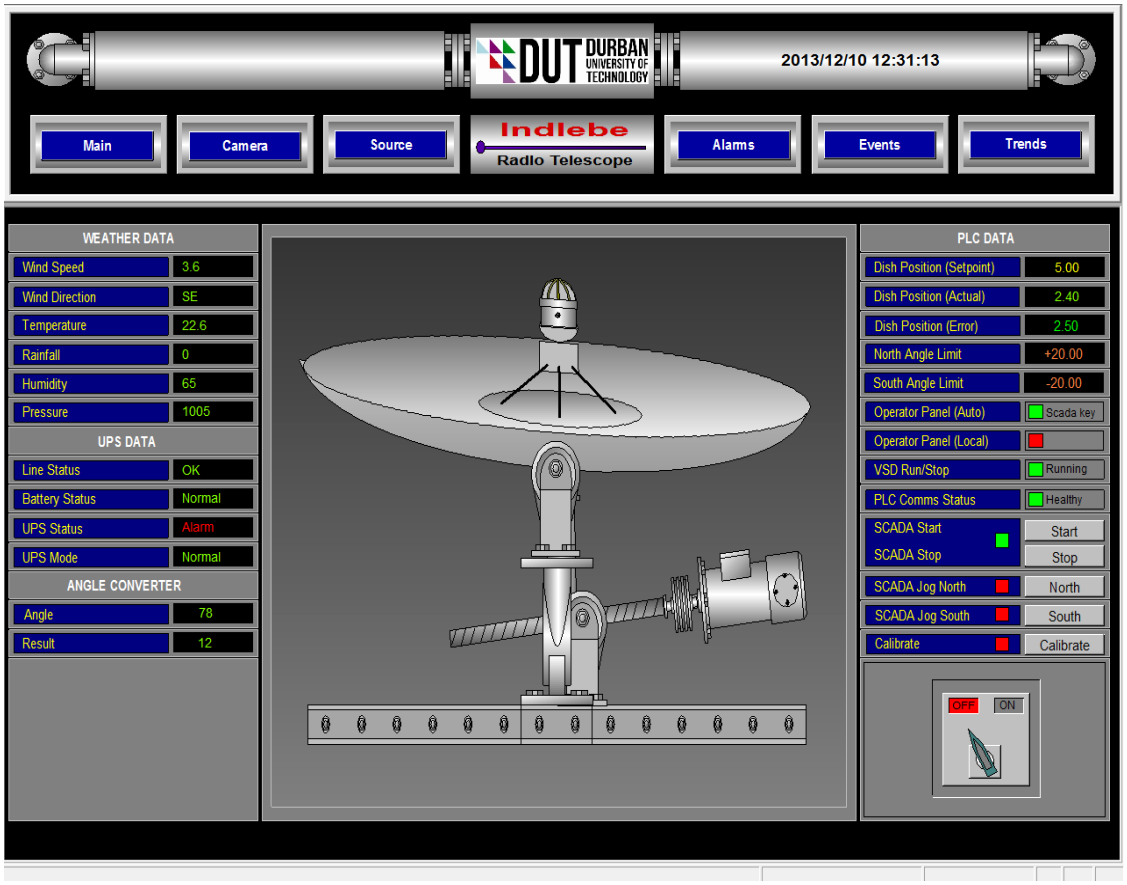
- Go to “User Account Control Settings”.
- Select “Start, Control Panel, User Accounts, Change User Account Control Settings – from “Default”, to “Never Notify”.
- Click on “Ok”.

APPENDIX B

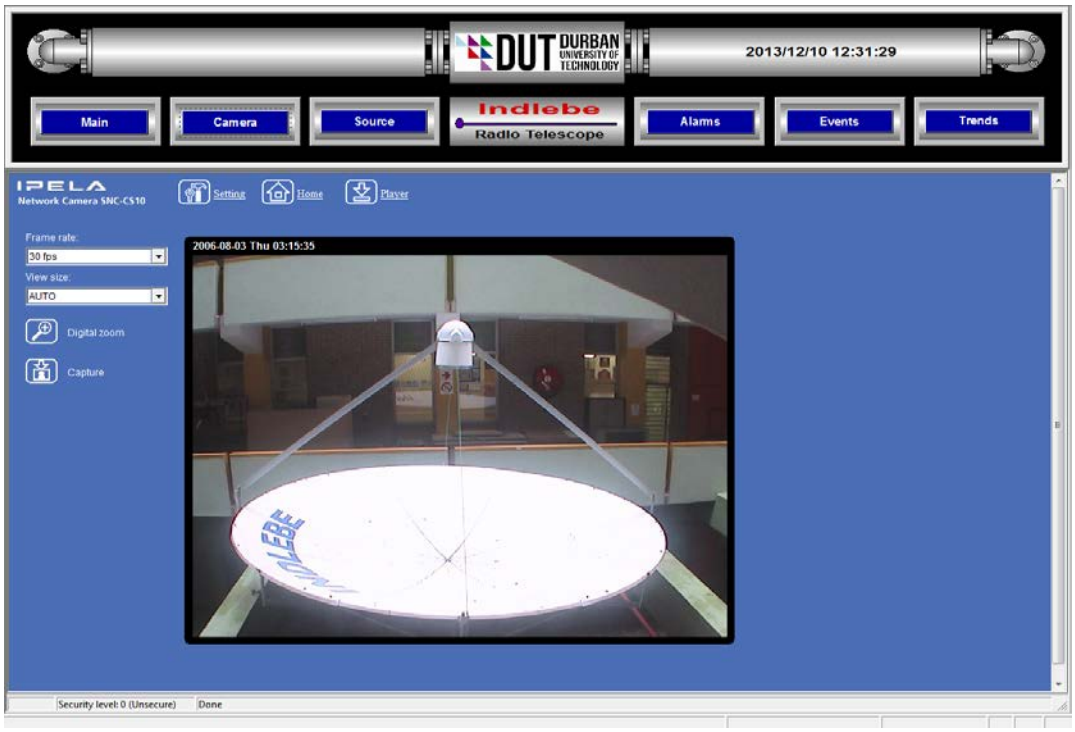
SCADA SCREEN LAYOUTS

Various SCADA screens have been developed to illustrate important information pertaining to this project. The SCADA system has a main header that can be used to navigate to the different screen options. The layout of each screen is illustrated below:

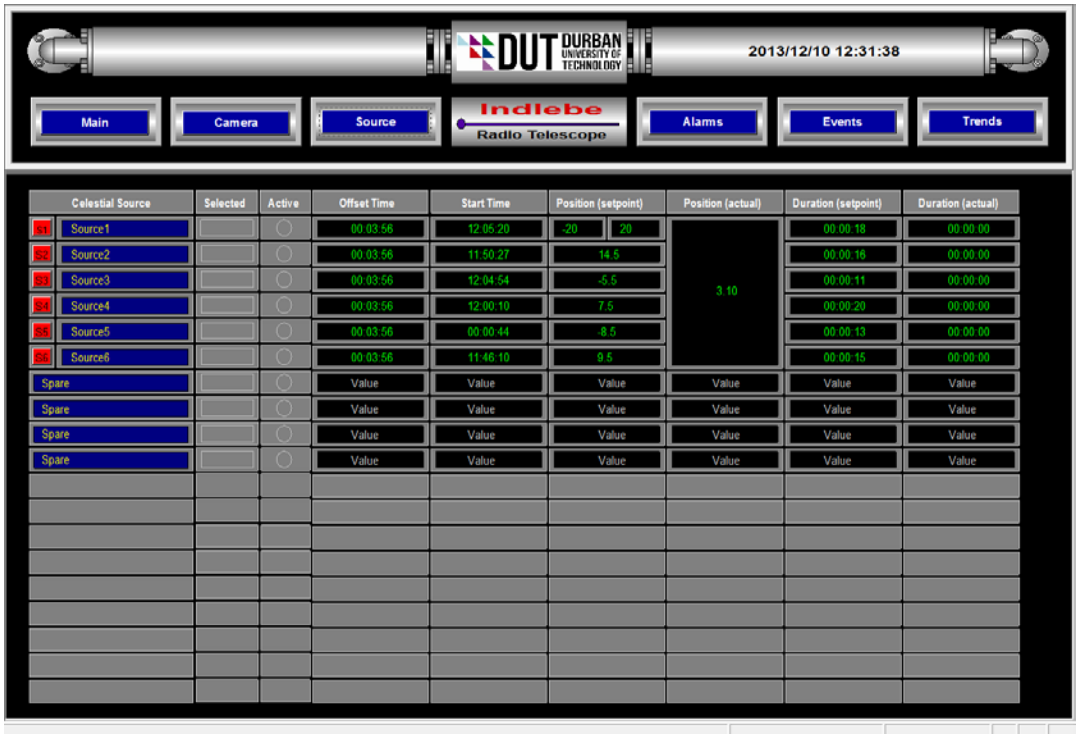
- SCADA Main Mimic



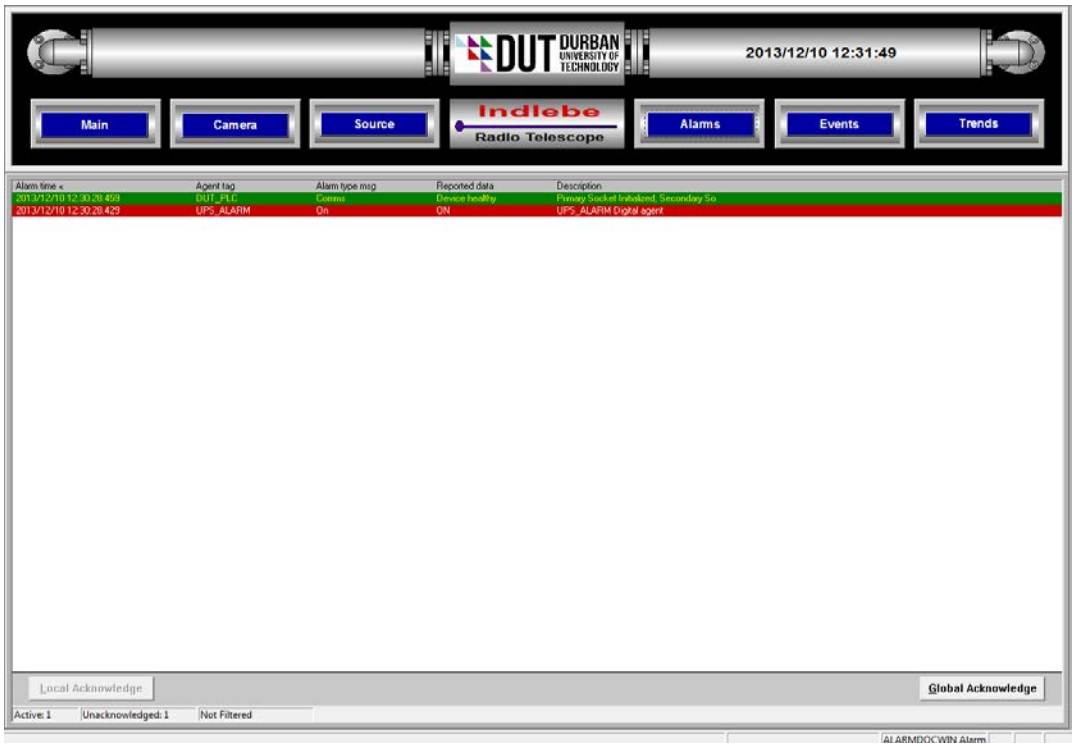
- SCADA Camera Mimic



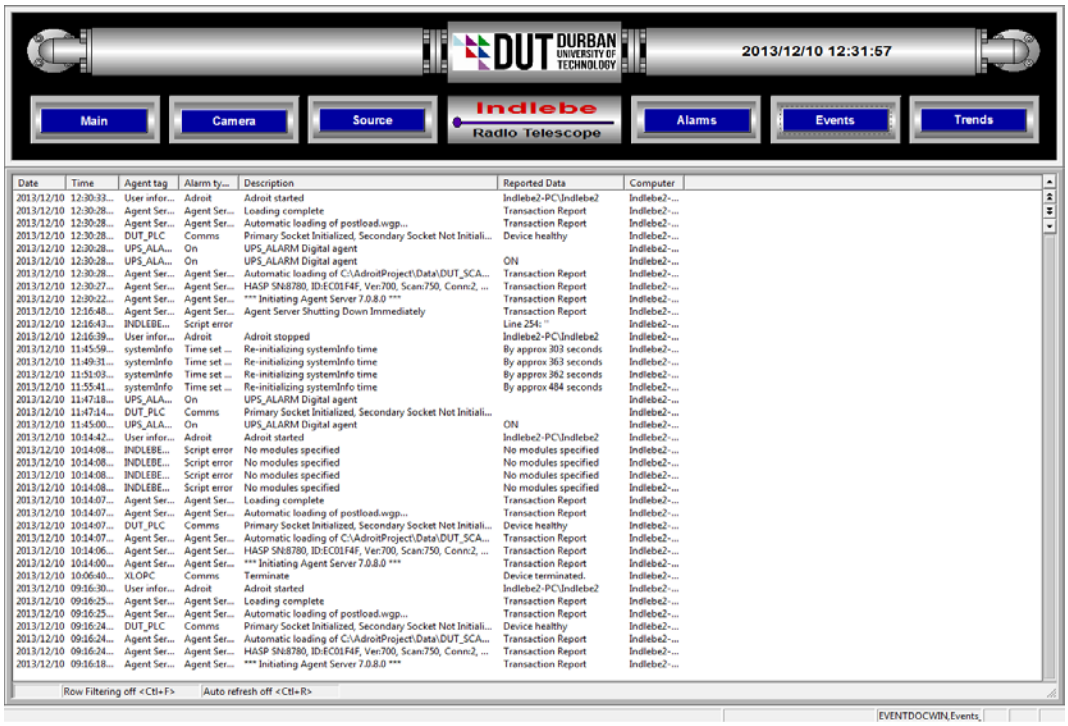
- SCADA Source Mimic



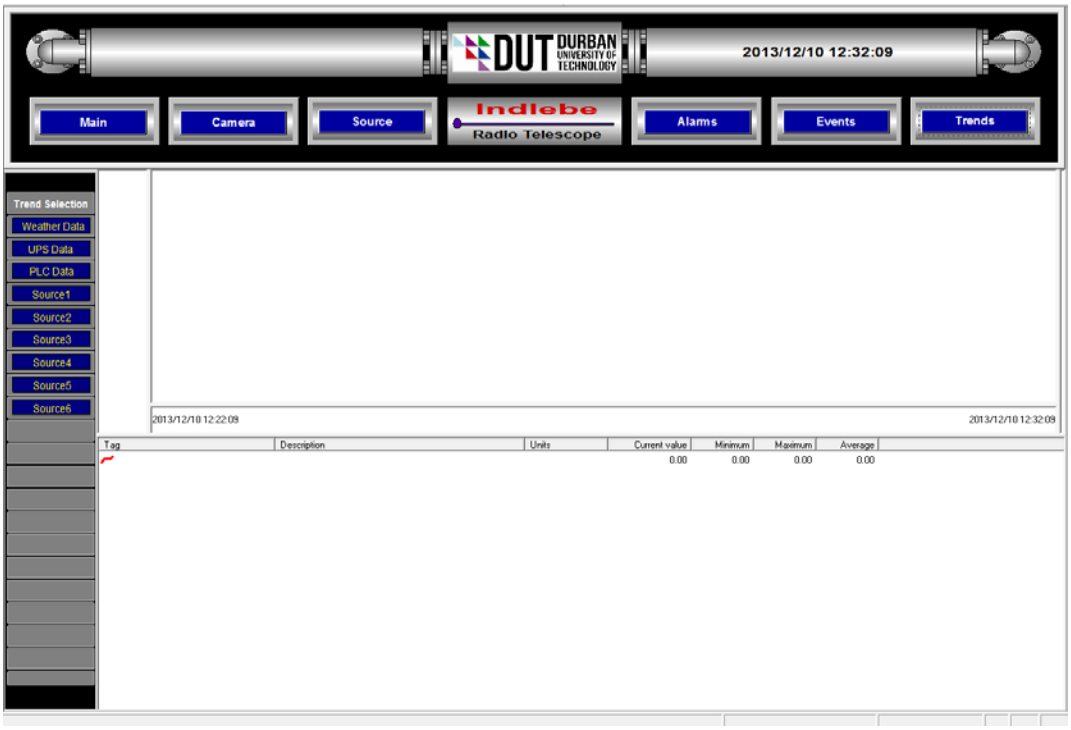
- SCADA Alarms Mimic



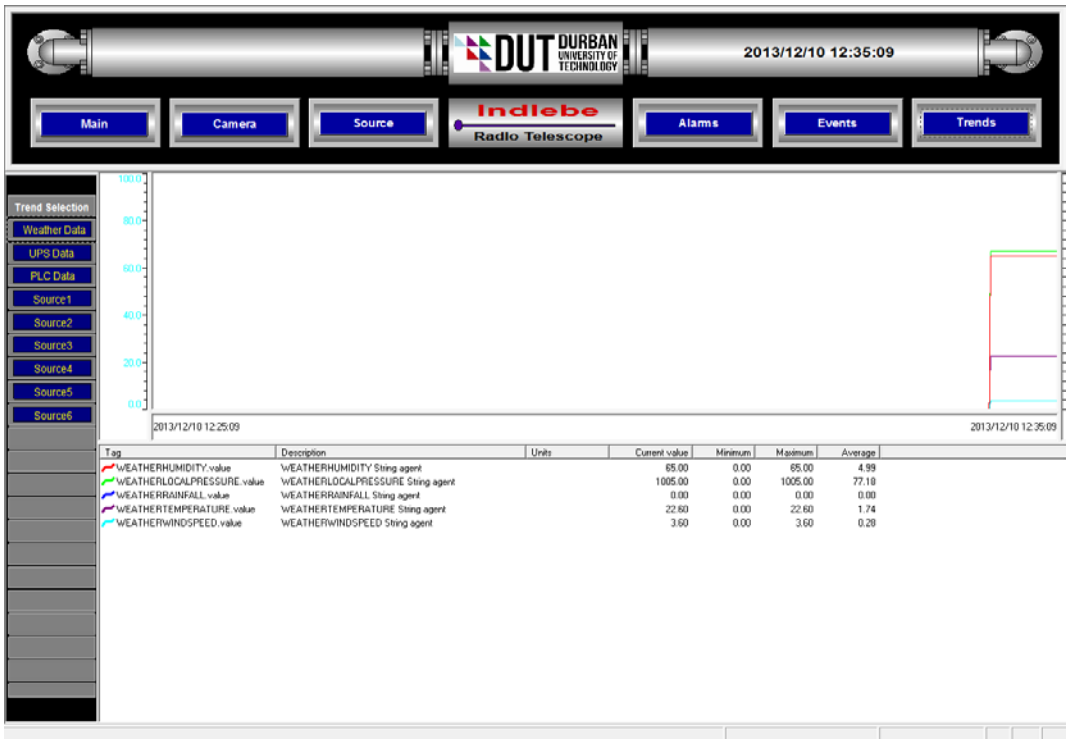
- SCADA Events Mimic



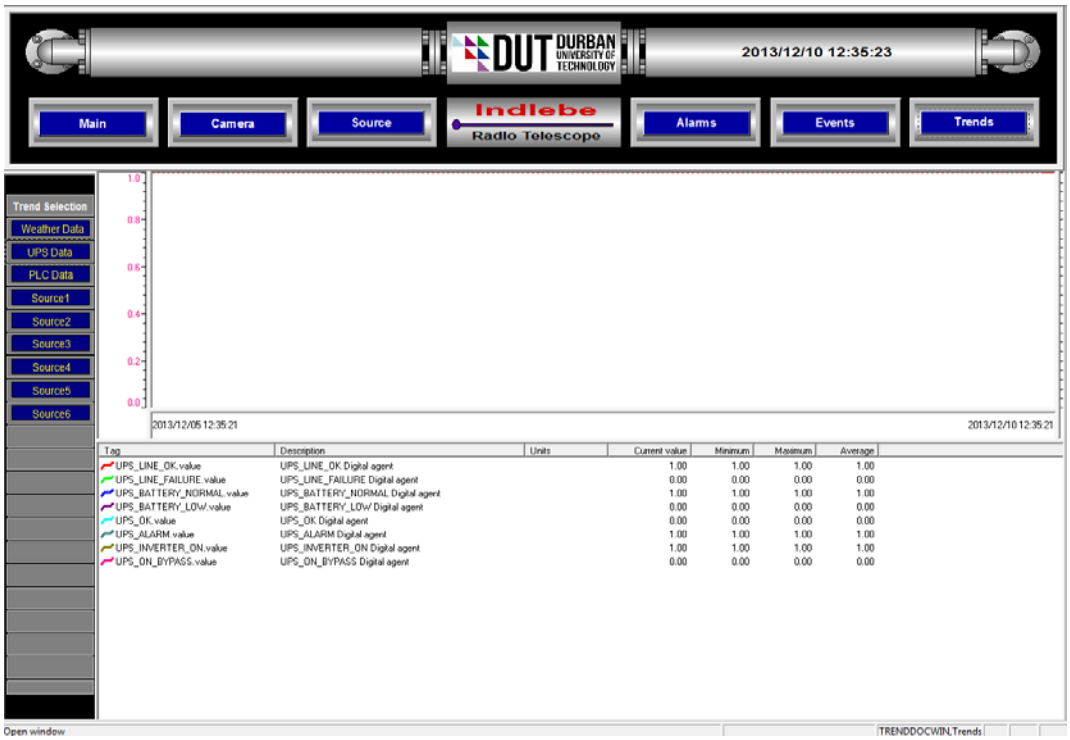
- SCADA Trends Mimic



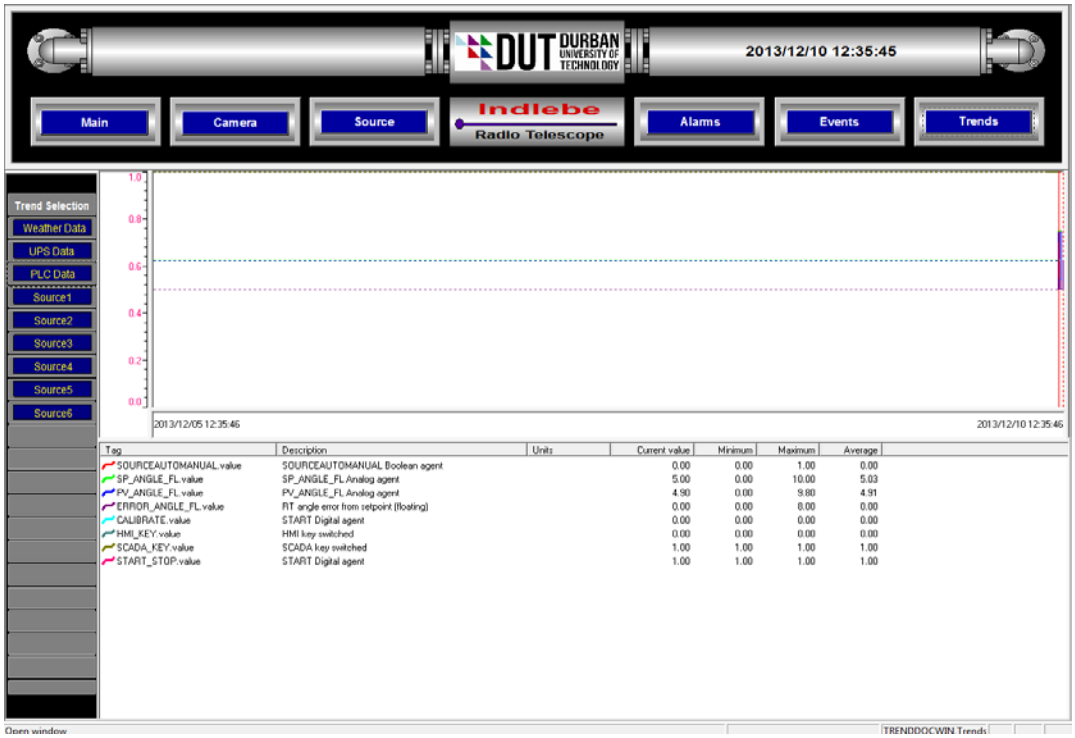
- SCADA Trends – Weather Data Mimic



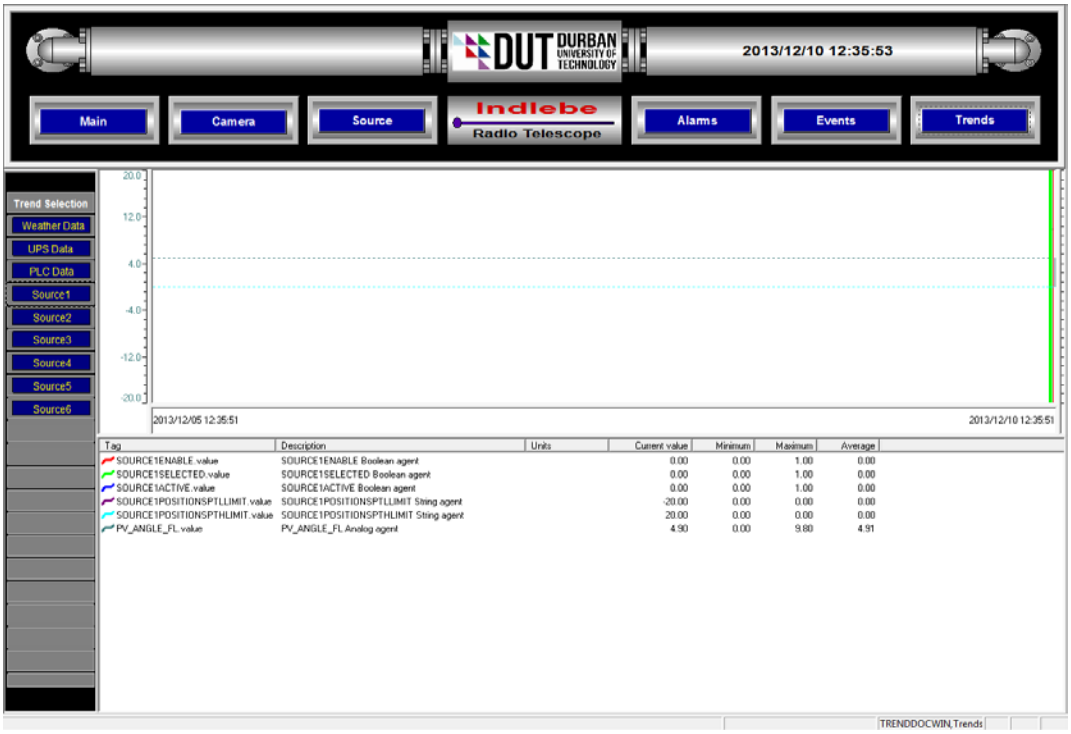
- SCADA Trends – UPS Data Mimic



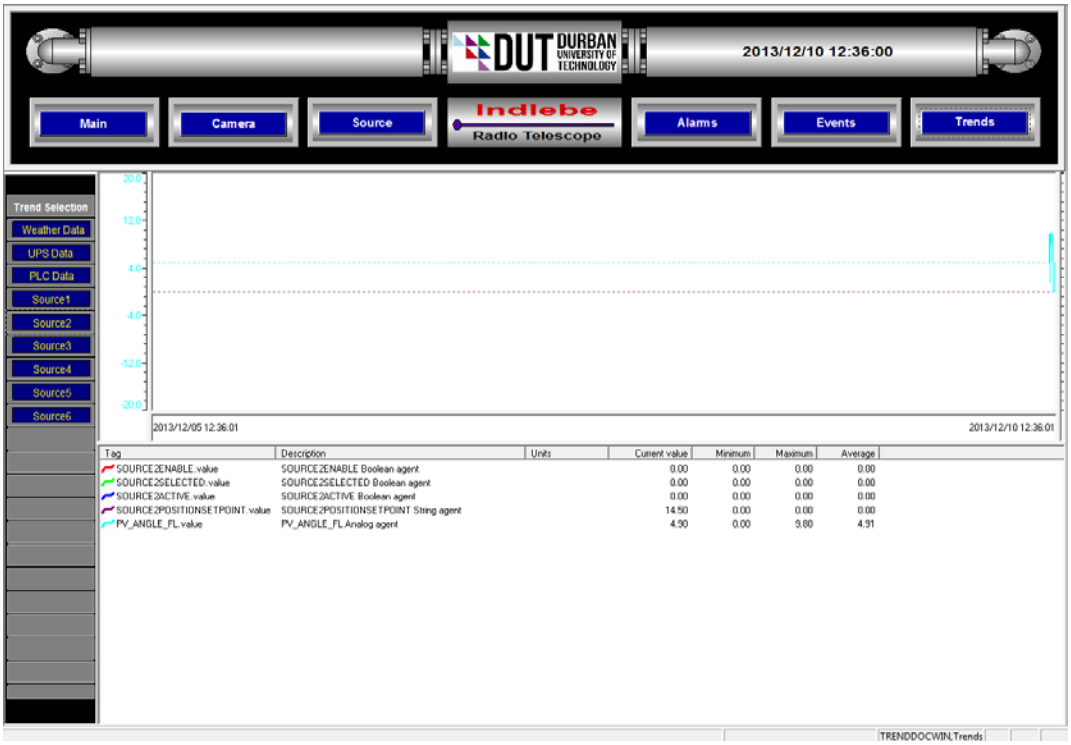
- SCADA Trends – PLC Data Mimic



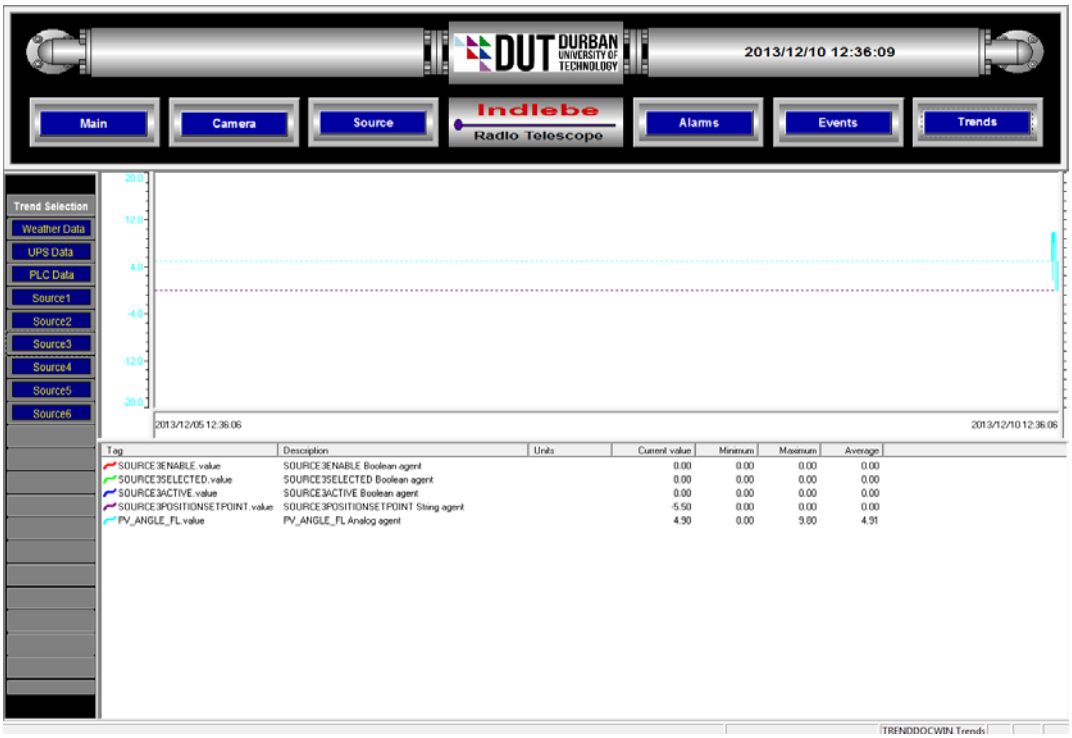
- SCADA Trends – Source1 Data Mimic



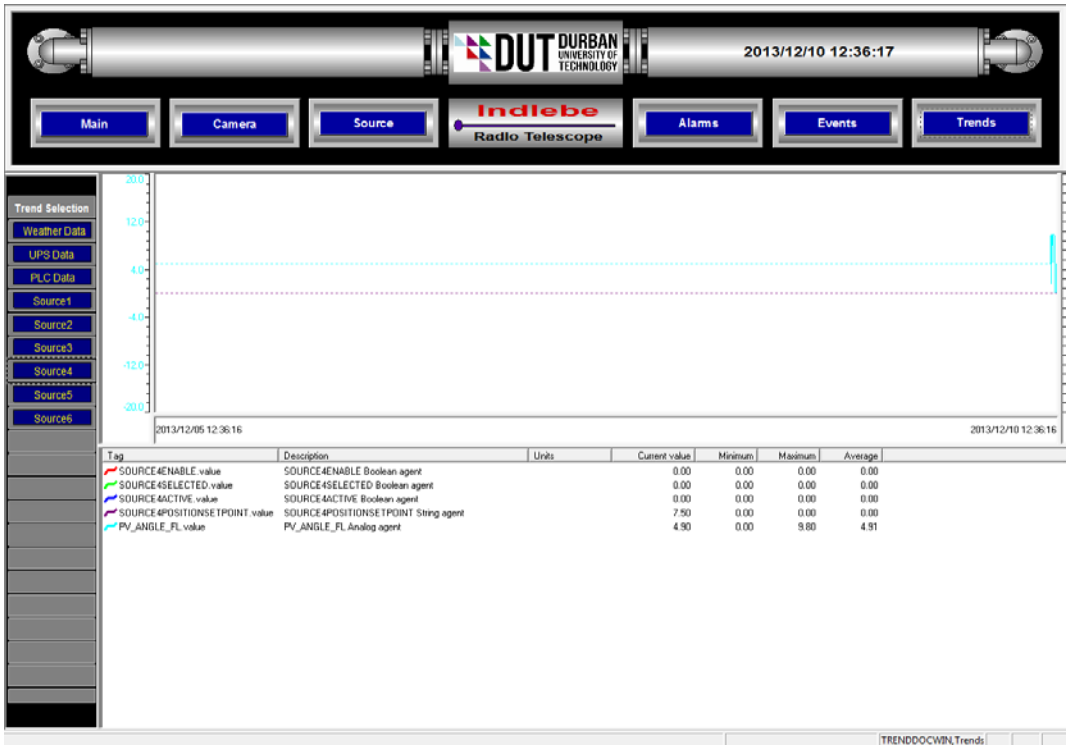
- SCADA Trends – Source2 Data Mimic



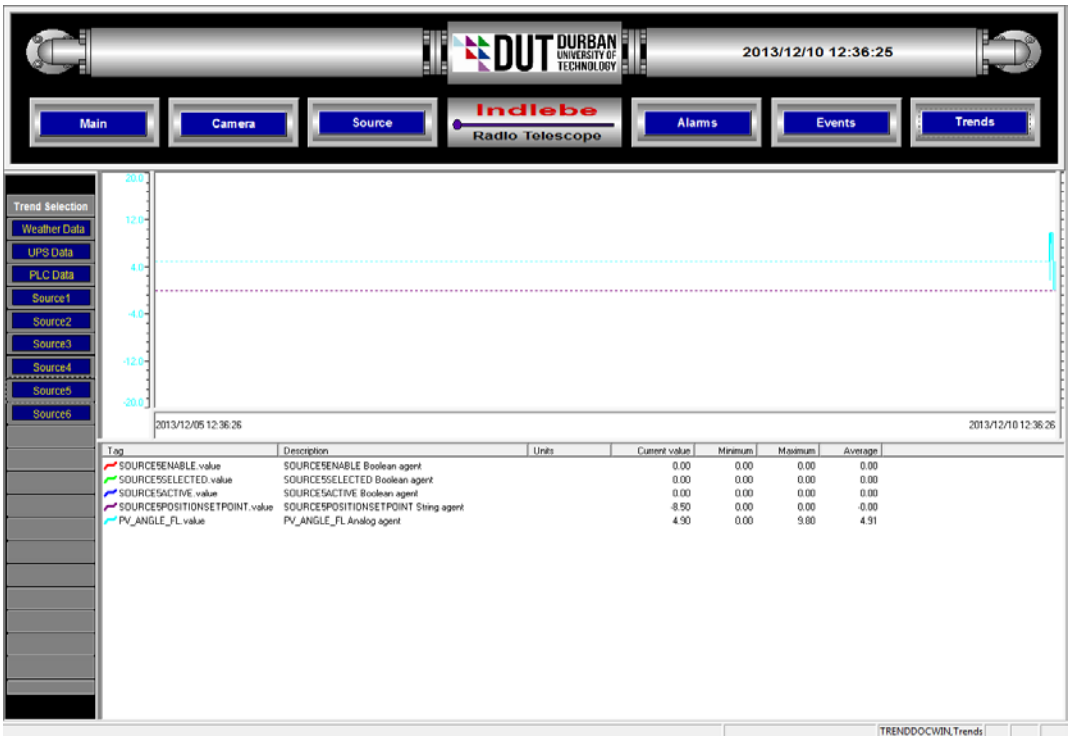
- SCADA Trends – Source3 Data Mimic



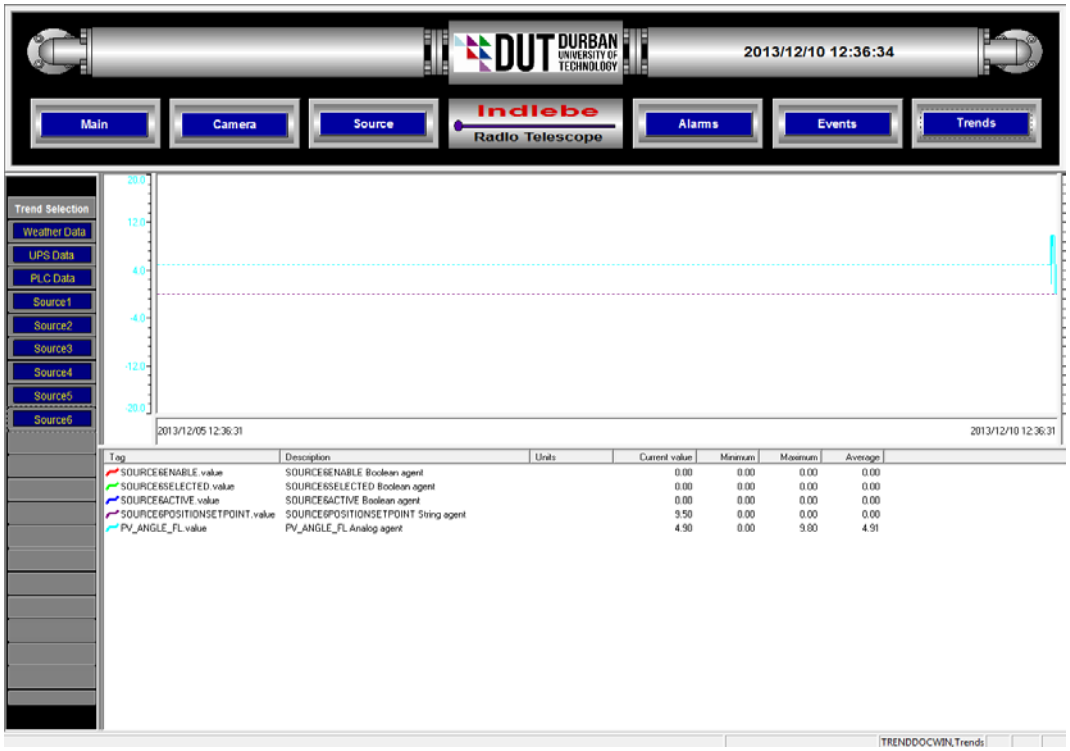
- SCADA Trends – Source4 Data Mimic



- SCADA Trends – Source5 Data Mimic



- SCADA Trends – Source6 Data Mimic



APPENDIX C

SCADA AUTOMATIC AND MANUAL CONTROL SCRIPT AND FUNCTIONALITY

```
Sub Main()


'Main software.....
'General variable declaration
Dim Sourceautomanual
Dim Sourcemanualpositionsetpointrequest,  SourcemanualpositionsetpointtoPLC,
      Sourcemanualpositionsetpointcheck
Dim Sourcemanualpositionsetpointrequestreset
Dim Dishmaxposition, Dishminposition
Dim Actualtime
Dim Priority1time, Priority2time, Priority3time, Priority4time, Priority5time, Priority6time
Dim  Priority1timeselected,  Priority2timeselected,  Priority3timeselected,  Priority4timeselected,
     Priority5timeselected, Priority6timeselected
Dim Sourcetotalenable
Dim SourceautopositionsetpointtoPLC
Dim Sourcepositionactualatsetpoint
Dim SourcepositionactualfromPLC


'Source1 varibale declaration
Dim Source1enable
Dim Source1name, Source1namerequest, Source1namerequestreset
```

```
Dim Source1positionscansetpoint,Source1postionscansetpointminusone,
    Source1positionscandaycount
Dim Source1active
Dim Source1offsettime, Source1offsettimerequest, Source1offsettimerequestreset
Dim Source1starttime,Source1starttimerequest,Source1starttimerequestreset,
      Source1starttimeconvert
Dim Source1starttimediff
Dim Source1starttimeminusoffsettime, Source1offsettimeretrieve, Source1starttimeretrieve
Dim Source1starttimeoffsettimedifference,   Source1starttimeoffsettimecompare1,
    Source1starttimeoffsettimecompare2, Source1starttimeoffsettimecompare3
Dim Source1positionsetpoint,Source1positionsetpointlowlimit,
      Source1positionsetpointlowlimitrequest,Source1positionsetpointlowlimitrequestreset,
    Source1positionsetpointhighlimit,Source1positionsetpointhighlimitrequest,
    Source1positionsetpointhighlimitrequestreset, Source1positionsetpointtoPLC
Dim Source1durationsetpoint,Source1durationsetpointrequest,
      Source1durationsetpointrequestreset, Source1durationactual
Dim Source1durationsetpointretrieve, Source1actualdurationatsetpoint
Dim Source1testtime
Dim Source1Source2compare,Source1Source3compare,Source1Source4compare,
    Source1Source5compare, Source1Source6compare
Dim Source1priority
Dim Source1priority1selected,Source1priority2selected,Source1priority3selected,
    Source1priority4selected, Source1priority5selected, Source1priority6selected
Dim Source1sequencecompleted
```

```
'Source2 variable declaration
Dim Source2enable
Dim Source2name, Source2namerequest, Source2namerequestreset
Dim Source2active
Dim Source2offsettime, Source2offsettimerequest, Source2offsettimerequestreset
Dim Source2starttime,Source2starttimerequest,Source2starttimerequestreset,
    Source2starttimeconvert
Dim Source2starttimediff
Dim Source2starttimeminusoffsettime, Source2offsettimeretrieve, Source2starttimeretrieve
Dim Source2starttimeoffsettimedifference,Source2starttimeoffsettimecompare1,
    Source2starttimeoffsettimecompare2, Source2starttimeoffsettimecompare3
Dim Source2positionsetpoint,Source2positionsetpointrequest,
    Source2positionsetpointrequestreset, Source2positionsetpointtoPLC
Dim Source2durationsetpoint,Source2durationsetpointrequest,
    Source2durationsetpointrequestreset, Source2durationactual
Dim Source2durationsetpointretrieve, Source2actualdurationatsetpoint
Dim Source2testtime
Dim Source2Source1compare,Source2Source3compare,Source2Source4compare,
    Source2Source5compare, Source2Source6compare
Dim Source2priority
Dim Source2priority1selected,Source2priority2selected,Source2priority3selected,
    Source2priority4selected, Source2priority5selected, Source2priority6selected
Dim Source2sequencecompleted
```

```
'Source3 variable declaration
Dim Source3enable
Dim Source3name, Source3namerequest, Source3namerequestreset
Dim Source3active
Dim Source3offsettime, Source3offsettimerequest, Source3offsettimerequestreset
Dim Source3starttime,Source3starttimerequest,Source3starttimerequestreset,
    Source3starttimeconvert
Dim Source3starttimediff
Dim Source3starttimeminusoffsettime, Source3offsettimeretrieve, Source3starttimeretrieve
Dim Source3starttimeoffsettimedifference,Source3starttimeoffsettimecompare1,
    Source3starttimeoffsettimecompare2, Source3starttimeoffsettimecompare3
Dim Source3positionsetpoint,Source3positionsetpointrequest,
    Source3positionsetpointrequestreset, Source3positionsetpointtoPLC
Dim Source3durationsetpoint,Source3durationsetpointrequest,
    Source3durationsetpointrequestreset, Source3durationactual
Dim Source3durationsetpointretrieve, Source3actualdurationatsetpoint
Dim Source3testtime
Dim Source3Source1compare,Source3Source2compare,Source3Source4compare,
    Source3Source5compare, Source3Source6compare
Dim Source3priority
Dim Source3priority1selected,Source3priority2selected,Source3priority3selected,
    Source3priority4selected, Source3priority5selected, Source3priority6selected
Dim Source3sequencecompleted
```

```
'Source4 variable declaration
Dim Source4enable
Dim Source4name, Source4namerequest, Source4namerequestreset
Dim Source4active
Dim Source4offsettime, Source4offsettimerequest, Source4offsettimerequestreset
Dim Source4starttime,Source4starttimerequest,Source4starttimerequestreset,
    Source4starttimeconvert
Dim Source4starttimediff
Dim Source4starttimeminusoffsettime, Source4offsettimeretrieve, Source4starttimeretrieve
Dim Source4starttimeoffsettimedifference,Source4starttimeoffsettimecompare1,
    Source4starttimeoffsettimecompare2, Source4starttimeoffsettimecompare3
Dim Source4positionsetpoint,Source4positionsetpointrequest,
    Source4positionsetpointrequestreset, Source4positionsetpointtoPLC
Dim Source4durationsetpoint,Source4durationsetpointrequest,
    Source4durationsetpointrequestreset, Source4durationactual
Dim Source4durationsetpointretrieve, Source4actualdurationatsetpoint
Dim Source4testtime
Dim Source4Source1compare,Source4Source2compare,Source4Source3compare,
    Source4Source5compare, Source4Source6compare
Dim Source4priority
Dim Source4priority1selected,Source4priority2selected,Source4priority3selected,
    Source4priority4selected, Source4priority5selected, Source4priority6selected
Dim Source4sequencecompleted
```

```
'Source5 variable declaration
Dim Source5enable
Dim Source5name, Source5namerequest, Source5namerequestreset
Dim Source5active
Dim Source5offsettime, Source5offsettimerequest, Source5offsettimerequestreset
Dim Source5starttime,Source5starttimerequest,Source5starttimerequestreset,
    Source5starttimeconvert
Dim Source5starttimediff
Dim Source5starttimeminusoffsettime, Source5offsettimeretrieve, Source5starttimeretrieve
Dim Source5starttimeoffsettimedifference,Source5starttimeoffsettimecompare1,
    Source5starttimeoffsettimecompare2, Source5starttimeoffsettimecompare3
Dim Source5positionsetpoint,Source5positionsetpointrequest,
    Source5positionsetpointrequestreset, Source5positionsetpointtoPLC
Dim Source5durationsetpoint,Source5durationsetpointrequest,
    Source5durationsetpointrequestreset, Source5durationactual
Dim Source5durationsetpointretrieve, Source5actualdurationatsetpoint
Dim Source5testtime
Dim Source5Source1compare,Source5Source2compare,Source5Source3compare,
    Source5Source4compare, Source5Source6compare
Dim Source5priority
Dim Source5priority1selected,Source5priority2selected,Source5priority3selected,
    Source5priority4selected, Source5priority5selected, Source5priority6selected
Dim Source5sequencecompleted
```

```
'Source6 variable declaration
Dim Source6enable
Dim Source6name, Source6namerequest, Source6namerequestreset
Dim Source6active
Dim Source6offsettime, Source6offsettimerequest, Source6offsettimerequestreset
Dim Source6starttime,Source6starttimerequest,Source6starttimerequestreset,
    Source6starttimeconvert
Dim Source6starttimediff
Dim Source6starttimeminusoffsettime, Source6offsettimeretrieve, Source6starttimeretrieve
Dim Source6starttimeoffsettimedifference,Source6starttimeoffsettimecompare1,
    Source6starttimeoffsettimecompare2, Source6starttimeoffsettimecompare3
Dim Source6positionsetpoint,Source6positionsetpointrequest,
    Source6positionsetpointrequestreset, Source6positionsetpointtoPLC
Dim Source6durationsetpoint,Source6durationsetpointrequest,
    Source6durationsetpointrequestreset, Source6durationactual
Dim Source6durationsetpointretrieve, Source6actualdurationatsetpoint
Dim Source6testtime
Dim Source6Source1compare,Source6Source2compare,Source6Source3compare,
    Source6Source4compare, Source6Source6compare
Dim Source6priority
Dim Source6priority1selected,Source6priority2selected,Source6priority3selected,
    Source6priority4selected, Source6priority5selected, Source6priority6selected
Dim Source6sequencecompleted
```

```
'General data transfer
    Sourceautomanual = Adroit.GetTag ("SOURCEAUTOMANUAL.value")
    Sourcemanualpositionsetpointrequest = Adroit.GetTag
        ("SOURCEMANUALPOSITIONSPTREQ.value")
    Dishmaxposition = 20
    Dishminposition = -20
    Toleranceplus = 0.5
    Toleranceminus = 0.5
    Inputboxxposition = 5000 'position for DUT is 7000, position for laptop is 5000
    Inputboxyposition = 8000 'position for DUT is 11000, position for laptop is 8000
    Roundtodecimalplace = 1 'round a number to a number of decimal places

'Manual software program.....
'Manual enabled
Do While Sourceautomanual = false

'Source1 enable reset
   Adroit.SetTag "SOURCE1ENABLE.value", 0
   Adroit.SetTag "SOURCE1SELECTED.value", false
   Source1active = false
   Adroit.SetTag "SOURCE1ACTIVE.value", Source1active
   Source1durationactual = TimeSerial (00,00,00)
   Adroit.SetTag "SOURCE1DURATIONACTUAL.value" , Source1durationactual
```

```
Sourcepositionactualatsetpoint = false
Source1actualdurationatsetpoint = false
Source1starttimeminusoffsettime = false
Source1starttimeoffsettimecompare3 = false

Source1postionscansetpointminusone = false
Source1sequencecompleted = true
Source2sequencecompleted = false
Source3sequencecompleted = false
Source4sequencecompleted = false
Source5sequencecompleted = false
Source6sequencecompleted = false
Source1namerequestreset = 0
Adroit.SetTag "SOURCE1NAMEREQ.value", Source1namerequestreset
Source1offsettimerequestreset = 0
Adroit.SetTag "SOURCE1OFFSETTIMEREQ.value", Source1offsettimerequestreset
Source1starttimerequestreset = 0
Adroit.SetTag "SOURCE1STARTTIMEREQ.value", Source1starttimerequestreset
Source1positionsetpointlowlimit = -20 'load a default of -20 to Source1 position setpoint low
    limit
Adroit.SetTag "SOURCE1POSITIONSPTLLIMIT.value" , Source1positionsetpointlowlimit
Source1positionsetpointhighlimit = 20 'load a default of 20 to Source1 position setpoint high
    limit
Adroit.SetTag "SOURCE1POSITIONSPTHLIMIT.value" , Source1positionsetpointhighlimit
```

```
    Source1positionsetpointlowlimitrequest = 0
    Adroit.SetTag"SOURCE1POSITIONSPTLLIMITREQ.value",
        Source1positionsetpointlowlimitrequest
    Source1positionsetpointhighlimitrequest = 0
    Adroit.SetTag"SOURCE1POSITIONSPTHLIMITREQ.value",
        Source1positionsetpointhighlimitrequest
    Source1durationsetpointrequestreset = 0
    Adroit.SetTag"SOURCE1DURATIONSETPOINTREQ.value",
        Source1durationsetpointrequestreset

'Source2 enable reset
    Adroit.SetTag "SOURCE2ENABLE.value", 0
    Adroit.SetTag "SOURCE2SELECTED.value", false
    Source2active = false
    Adroit.SetTag "SOURCE2ACTIVE.value", Source2active
    Source2durationactual = TimeSerial (00,00,00)
    Adroit.SetTag "SOURCE2DURATIONACTUAL.value" , Source2durationactual
    Sourcepositionactualatsetpoint = false
    Source2actualdurationatsetpoint = false
    Source2starttimeminusoffsettime = false
    Source2starttimeoffsettimecompare3 = false
    Source2sequencecompleted = true
    Source1sequencecompleted = false
    Source3sequencecompleted = false
```

```
    Source4sequencecompleted = false
    Source5sequencecompleted = false
    Source6sequencecompleted = false
    Source2namerequestreset = 0
    Adroit.SetTag "SOURCE2NAMEREQ.value", Source2namerequestreset
    Source2offsettimerequestreset = 0
    Adroit.SetTag "SOURCE2OFFSETTIMEREQ.value", Source2offsettimerequestreset
    Source2starttimerequestreset = 0
    Adroit.SetTag "SOURCE2STARTTIMEREQ.value", Source2starttimerequestreset
    Source2positionsetpointrequestreset = 0
    Adroit.SetTag"SOURCE2POSITIONSETPOINTREQ.value",
        Source2positionsetpointrequestreset
    Source2durationsetpointrequestreset = 0
    Adroit.SetTag"SOURCE2DURATIONSETPOINTREQ.value",
        Source2durationsetpointrequestreset

'Source3 enable reset
    Adroit.SetTag "SOURCE3ENABLE.value", 0
    Adroit.SetTag "SOURCE3SELECTED.value", false
    Source3active = false
    Adroit.SetTag "SOURCE3ACTIVE.value", Source3active
    Source3durationactual = TimeSerial (00,00,00)
    Adroit.SetTag "SOURCE3DURATIONACTUAL.value" , Source3durationactual
    Sourcepositionactualatsetpoint = false
```

```
    Source3actualdurationatsetpoint = false
    Source3starttimeminusoffsettime = false
    Source3starttimeoffsettimecompare3 = false
    Source3sequencecompleted = true
    Source1sequencecompleted = false
    Source2sequencecompleted = false
    Source4sequencecompleted = false
    Source5sequencecompleted = false
    Source6sequencecompleted = false
    Source3namerequestreset = 0
    Adroit.SetTag "SOURCE3NAMEREQ.value", Source3namerequestreset
    Source3offsettimerequestreset = 0
    Adroit.SetTag "SOURCE3OFFSETTIMEREQ.value", Source3offsettimerequestreset
    Source3starttimerequestreset = 0
    Adroit.SetTag "SOURCE3STARTTIMEREQ.value", Source3starttimerequestreset
    Source3positionsetpointrequestreset = 0
    Adroit.SetTag"SOURCE3POSITIONSETPOINTREQ.value",
        Source3positionsetpointrequestreset
    Source3durationsetpointrequestreset = 0
    Adroit.SetTag"SOURCE3DURATIONSETPOINTREQ.value",
        Source3durationsetpointrequestreset

'Source4 enable reset
    Adroit.SetTag "SOURCE4ENABLE.value", 0
```

```
Adroit.SetTag "SOURCE4SELECTED.value", false

Source4active = false

Adroit.SetTag "SOURCE4ACTIVE.value", Source4active

Source4durationactual = TimeSerial (00,00,00)

Adroit.SetTag "SOURCE4DURATIONACTUAL.value" , Source4durationactual

Sourcepositionactualatsetpoint = false

Source4actualdurationatsetpoint = false

Source4starttimeminusoffsettime = false

Source4starttimeoffsettimecompare3 = false

Source4sequencecompleted = true

Source1sequencecompleted = false

Source2sequencecompleted = false

Source3sequencecompleted = false

Source5sequencecompleted = false

Source6sequencecompleted = false

Source4namerequestreset = 0

Adroit.SetTag "SOURCE4NAMEREQ.value", Source4namerequestreset

Source4offsettimerequestreset = 0

Adroit.SetTag "SOURCE4OFFSETTIMEREQ.value", Source4offsettimerequestreset

Source4starttimerequestreset = 0

Adroit.SetTag "SOURCE4STARTTIMEREQ.value", Source4starttimerequestreset

Source4positionsetpointrequestreset = 0

Adroit.SetTag"SOURCE4POSITIONSETPOINTREQ.value",

    Source4positionsetpointrequestreset
```

```
    Source4durationsetpointrequestreset = 0
    Adroit.SetTag"SOURCE4DURATIONSETPOINTREQ.value",
        Source4durationsetpointrequestreset

'Source5 enable reset
    Adroit.SetTag "SOURCE5ENABLE.value", 0
    Adroit.SetTag "SOURCE5SELECTED.value", false
    Source5active = false
    Adroit.SetTag "SOURCE5ACTIVE.value", Source5active
    Source5durationactual = TimeSerial (00,00,00)
    Adroit.SetTag "SOURCE5DURATIONACTUAL.value" , Source5durationactual
    Sourcepositionactualatsetpoint = false
    Source5actualdurationatsetpoint = false
    Source5starttimeminusoffsettime = false
    Source5starttimeoffsettimecompare3 = false
    Source5sequencecompleted = true
    Source1sequencecompleted = false
    Source2sequencecompleted = false
    Source3sequencecompleted = false
    Source4sequencecompleted = false
    Source6sequencecompleted = false
    Source5namerequestreset = 0
    Adroit.SetTag "SOURCE5NAMEREQ.value", Source5namerequestreset
    Source5offsettimerequestreset = 0
```

```
    Adroit.SetTag "SOURCE5OFFSETTIMEREQ.value", Source5offsettimerequestreset
    Source5starttimerequestreset = 0
    Adroit.SetTag "SOURCE5STARTTIMEREQ.value", Source5starttimerequestreset
    Source5positionsetpointrequestreset = 0
    Adroit.SetTag"SOURCE5POSITIONSETPOINTREQ.value",
        Source5positionsetpointrequestreset
    Source5durationsetpointrequestreset = 0
    Adroit.SetTag"SOURCE5DURATIONSETPOINTREQ.value",
        Source5durationsetpointrequestreset

'Source6 enable reset
    Adroit.SetTag "SOURCE6ENABLE.value", 0
    Adroit.SetTag "SOURCE6SELECTED.value", false
    Source6active = false
    Adroit.SetTag "SOURCE6ACTIVE.value", Source6active
    Source6durationactual = TimeSerial (00,00,00)
    Adroit.SetTag "SOURCE6DURATIONACTUAL.value" , Source6durationactual
    Sourcepositionactualatsetpoint = false
    Source6actualdurationatsetpoint = false
    Source6starttimeminusoffsettime = false
    Source6starttimeoffsettimecompare3 = false
    Source6sequencecompleted = true
    Source1sequencecompleted = false
    Source2sequencecompleted = false
```

```
    Source3sequencecompleted = false
    Source4sequencecompleted = false
    Source5sequencecompleted = false
    Source6namerequestreset = 0
    Adroit.SetTag "SOURCE6NAMEREQ.value", Source6namerequestreset
    Source6offsettimerequestreset = 0
    Adroit.SetTag "SOURCE6OFFSETTIMEREQ.value", Source6offsettimerequestreset
    Source6starttimerequestreset = 0
    Adroit.SetTag "SOURCE6STARTTIMEREQ.value", Source6starttimerequestreset
    Source6positionsetpointrequestreset = 0
    Adroit.SetTag"SOURCE6POSITIONSETPOINTREQ.value",
        Source6positionsetpointrequestreset
    Source6durationsetpointrequestreset = 0
    Adroit.SetTag"SOURCE6DURATIONSETPOINTREQ.value",
        Source6durationsetpointrequestreset

'General data transfer
    Sourceautomanual = Adroit.GetTag ("SOURCEAUTOMANUAL.value")
    SourcepositionactualfromPLC = Adroit.GetTag ("PV_ANGLE_FL.value")

If  SourcemanualpositionsetpointtoPLC  <=  Dishmaxposition  or
        SourcemanualpositionsetpointtoPLC >= Dishminposition Then
    Sourcemanualpositionsetpointrequest  =  Adroit.GetTag
        ("SOURCEMANUALPOSITIONSPTREQ.value")
```

```
If Sourcemanualpositionsetpointrequest = true Then
  SourcemanualpositionsetpointtoPLC = Round (CDbl (InputBox("Enter dish manual position
    setpoint (+20deg to -20deg)","","",Inputboxxposition,Inputboxyposition)),
    Roundtodecimalplace)
  Sourcemanualpositionsetpointcheck = IsNumeric (SourcemanualpositionsetpointtoPLC)
  Adroit.SetTag "SCADA_ANGLE_SP_EXP.v01" , SourcemanualpositionsetpointtoPLC

If SourcemanualpositionsetpointtoPLC > Dishmaxposition or
    SourcemanualpositionsetpointtoPLC < Dishminposition Then
  MsgBox ("Invalid Entry - Angle Between +20deg to -20deg")
End If

If SourcemanualpositionsetpointtoPLC > SourcepositionactualfromPLC and not
    SourcemanualpositionsetpointtoPLC > Dishmaxposition Then 'Source postion actual
    below setpoint
   Do While SourcemanualpositionsetpointtoPLC > SourcepositionactualfromPLC
   SourcepositionactualfromPLC = Adroit.GetTag ("PV_ANGLE_FL.value")
   Sourcemanualpositionsetpointrequestreset = 0
   Adroit.SetTag "SOURCEMANUALPOSITIONSPTREQ.value" ,
    Sourcemanualpositionsetpointrequestreset
   Sourcemanualpositionsetpointrequest = Adroit.GetTag
    ("SOURCEMANUALPOSITIONSPTREQ.value")
```

```
    If   (SourcepositionactualfromPLC + Toleranceplus) >= SourcemanualpositionsetpointtoPLC
      Then
    Exit Do
    End If
    Loop
End If

If    SourcemanualpositionsetpointtoPLC    <    SourcepositionactualfromPLC    and    not
      SourcemanualpositionsetpointtoPLC < Dishminposition Then  'Source postion actual above
      setpoint
    Do While SourcemanualpositionsetpointtoPLC < SourcepositionactualfromPLC
    SourcepositionactualfromPLC = Adroit.GetTag ("PV_ANGLE_FL.value")
    Sourcemanualpositionsetpointrequestreset = 0
    Adroit.SetTag    "SOURCEMANUALPOSITIONSPTREQ.value"    ,
      Sourcemanualpositionsetpointrequestreset
    Sourcemanualpositionsetpointrequest    =    Adroit.GetTag
      ("SOURCEMANUALPOSITIONSPTREQ.value")
    If (SourcepositionactualfromPLC - Toleranceminus) <= SourcemanualpositionsetpointtoPLC
      Then
    Exit Do
    End If
    Loop
End If
```

```
'Manual diabled
If Sourceautomanual = true Then
Exit Do
End If
End If
End if
Loop

'Automatic software program.....
'Automatic enabled
Do While Sourceautomanual = true

'General data transfer
  Sourcepositionactualfrom PLC = Adroit.GetTag ("PV_ANGLE_FL.value")
  Actualtime = CDate(Time)
  Sourcetotalenable = (Source1enable + Source2enable + Source3enable + Source4enable +
      Source5enable + Source6enable)
  Sourcemanualpositionsetpointrequestreset = 0
  Adroit.SetTag "SOURCEMANUALPOSITIONSPTREQ.value" ,
      Sourcemanualpositionsetpointrequestreset

'Source1 request data transfer
  Source1enable = Abs (CInt (Adroit.GetTag ("SOURCE1ENABLE.value")))
  Source1namerequest = Adroit.GetTag ("SOURCE1NAMEREQ.value")
```

```
Source1offsettimerequest = Adroit.GetTag ("SOURCE1OFFSETTIMEREQ.value")
Source1offsettimeretrieve = TimeValue (Adroit.GetTag ("SOURCE1OFFSETTIME.value"))
Source1starttimerequest = Adroit.GetTag ("SOURCE1STARTTIMEREQ.value")
Source1starttimeconvert = Adroit.GetTag ("SOURCE1STARTTIME.value")
Source1starttimeretrieve = TimeValue (Adroit.GetTag ("SOURCE1STARTTIME.value"))
Source1testtime = CDate(Source1starttimeconvert) 'to convert Source1 time from a string to
    time value
Source1positionsetpointlowlimitrequest = Adroit.GetTag
    ("SOURCE1POSITIONSPTLLIMITREQ.value")
Source1positionsetpointhighlimitrequest = Adroit.GetTag
    ("SOURCE1POSITIONSPTHLIMITREQ.value")
Source1positionsetpointtoPLC = CInt (Adroit.GetTag
    ("SOURCE1POSITIONSETPOINT.value"))
Source1durationsetpointrequest = Adroit.GetTag
    ("SOURCE1DURATIONSETPOINTREQ.value")
Source1durationsetpointretrieve = TimeValue (Adroit.GetTag
    ("SOURCE1DURATIONSETPOINT.value"))
Source1durationincrement = TimeSerial (00,00,01)

'Source2 request data transfer
Source2enable = Abs (CInt (Adroit.GetTag ("SOURCE2ENABLE.value")))
Source2namerequest = Adroit.GetTag ("SOURCE2NAMEREQ.value")
Source2offsettimerequest = Adroit.GetTag ("SOURCE2OFFSETTIMEREQ.value")
Source2offsettimeretrieve = TimeValue (Adroit.GetTag ("SOURCE2OFFSETTIME.value"))
```

```
Source2starttimerequest = Adroit.GetTag ("SOURCE2STARTTIMEREQ.value")
Source2starttimeconvert = Adroit.GetTag ("SOURCE2STARTTIME.value")
Source2starttimeretrieve = TimeValue (Adroit.GetTag ("SOURCE2STARTTIME.value"))
Source2testtime = CDate(Source2starttimeconvert) 'to convert Source2 time from a string to
    time value
Source2positionsetpointrequest = Adroit.GetTag
    ("SOURCE2POSITIONSETPOINTREQ.value")
Source2positionsetpointtoPLC = CInt (Adroit.GetTag
    ("SOURCE2POSITIONSETPOINT.value"))
Source2durationsetpointrequest = Adroit.GetTag
    ("SOURCE2DURATIONSETPOINTREQ.value")
Source2durationsetpointretrieve = TimeValue (Adroit.GetTag
    ("SOURCE2DURATIONSETPOINT.value"))
Source2durationincrement = TimeSerial (00,00,01)

'Source3 request data transfer
Source3enable = Abs (CInt (Adroit.GetTag ("SOURCE3ENABLE.value")))
Source3namerequest = Adroit.GetTag ("SOURCE3NAMEREQ.value")
Source3offsettimerequest = Adroit.GetTag ("SOURCE3OFFSETTIMEREQ.value")
Source3offsettimeretrieve = TimeValue (Adroit.GetTag ("SOURCE3OFFSETTIME.value"))
Source3starttimerequest = Adroit.GetTag ("SOURCE3STARTTIMEREQ.value")
Source3starttimeconvert = Adroit.GetTag ("SOURCE3STARTTIME.value")
Source3starttimeretrieve = TimeValue (Adroit.GetTag ("SOURCE3STARTTIME.value"))
```

```
Source3testtime = CDate(Source3starttimeconvert) 'to convert Source3 time from a string to
    time value
Source3positionsetpointrequest = Adroit.GetTag
    ("SOURCE3POSITIONSETPOINTREQ.value")
Source3positionsetpointtoPLC = CInt (Adroit.GetTag
    ("SOURCE3POSITIONSETPOINT.value"))
Source3durationsetpointrequest = Adroit.GetTag
    ("SOURCE3DURATIONSETPOINTREQ.value")
Source3durationsetpointretrieve = TimeValue (Adroit.GetTag
    ("SOURCE3DURATIONSETPOINT.value"))
Source3durationincrement = TimeSerial (00,00,01)

'Source4 requst data transfer
Source4enable = Abs (CInt (Adroit.GetTag ("SOURCE4ENABLE.value")))
Source4namerequest = Adroit.GetTag ("SOURCE4NAMEREQ.value")
Source4offsettimerequest = Adroit.GetTag ("SOURCE4OFFSETTIMEREQ.value")
Source4offsettimeretrieve = TimeValue (Adroit.GetTag ("SOURCE4OFFSETTIME.value"))
Source4starttimerequest = Adroit.GetTag ("SOURCE4STARTTIMEREQ.value")
Source4starttimeconvert = Adroit.GetTag ("SOURCE4STARTTIME.value")
Source4starttimeretrieve = TimeValue (Adroit.GetTag ("SOURCE4STARTTIME.value"))
Source4testtime = CDate(Source4starttimeconvert) 'to convert Source4 time from a string to
    time value
Source4positionsetpointrequest = Adroit.GetTag
    ("SOURCE4POSITIONSETPOINTREQ.value")
```

```
  Source4positionsetpointtoPLC = CInt (Adroit.GetTag
     ("SOURCE4POSITIONSETPOINT.value"))
  Source4durationsetpointrequest = Adroit.GetTag
     ("SOURCE4DURATIONSETPOINTREQ.value")
  Source4durationsetpointretrieve = TimeValue (Adroit.GetTag
     ("SOURCE4DURATIONSETPOINT.value"))
  Source4durationincrement = TimeSerial (00,00,01)

'Source5 request data transfer
  Source5enable = Abs (CInt (Adroit.GetTag ("SOURCE5ENABLE.value")))
  Source5namerequest = Adroit.GetTag ("SOURCE5NAMEREQ.value")
  Source5offsettimerequest = Adroit.GetTag ("SOURCE5OFFSETTIMEREQ.value")
  Source5offsettimeretrieve = TimeValue (Adroit.GetTag ("SOURCE5OFFSETTIME.value"))
  Source5starttimerequest = Adroit.GetTag ("SOURCE5STARTTIMEREQ.value")
  Source5starttimeconvert = Adroit.GetTag ("SOURCE5STARTTIME.value")
  Source5starttimeretrieve = TimeValue (Adroit.GetTag ("SOURCE5STARTTIME.value"))
  Source5testtime = CDate(Source5starttimeconvert) 'to convert Source5 time from a string to
     time value
  Source5positionsetpointrequest = Adroit.GetTag
     ("SOURCE5POSITIONSETPOINTREQ.value")
  Source5positionsetpointtoPLC = CInt (Adroit.GetTag
     ("SOURCE5POSITIONSETPOINT.value"))
  Source5durationsetpointrequest = Adroit.GetTag
     ("SOURCE5DURATIONSETPOINTREQ.value")
```

```
Source5durationsetpointretrieve = TimeValue (Adroit.GetTag
    ("SOURCE5DURATIONSETPOINT.value"))
Source5durationincrement = TimeSerial (00,00,01)

'Source6 request data transfer
Source6enable = Abs (CInt (Adroit.GetTag ("SOURCE6ENABLE.value")))
Source6namerequest = Adroit.GetTag ("SOURCE6NAMEREQ.value")
Source6offsettimerequest = Adroit.GetTag ("SOURCE6OFFSETTIMEREQ.value")
Source6offsettimeretrieve = TimeValue (Adroit.GetTag ("SOURCE6OFFSETTIME.value"))
Source6starttimerequest = Adroit.GetTag ("SOURCE6STARTTIMEREQ.value")
Source6starttimeconvert = Adroit.GetTag ("SOURCE6STARTTIME.value")
Source6starttimeretrieve = TimeValue (Adroit.GetTag ("SOURCE6STARTTIME.value"))
Source6testtime = CDate(Source6starttimeconvert) 'to convert Source6 time from a string to
    time value
Source6positionsetpointrequest = Adroit.GetTag
    ("SOURCE6POSITIONSETPOINTREQ.value")
Source6positionsetpointtoPLC = CInt (Adroit.GetTag
    ("SOURCE6POSITIONSETPOINT.value"))
Source6durationsetpointrequest = Adroit.GetTag
    ("SOURCE6DURATIONSETPOINTREQ.value")
Source6durationsetpointretrieve = TimeValue (Adroit.GetTag
    ("SOURCE6DURATIONSETPOINT.value"))
Source6durationincrement = TimeSerial (00,00,01)
```

```
'Source1 Enable
If Source1enable = 1 Then
'Source1 data request reset
Adroit.SetTag "SOURCE1NAMEREQ.value", false
Adroit.SetTag "SOURCE1OFFSETTIMEREQ.value", false
Adroit.SetTag "SOURCE1STARTTIMEREQ.value", false
Adroit.SetTag "SOURCE1POSITIONSPTLLIMITREQ.value", false
Adroit.SetTag "SOURCE1POSITIONSPTHLIMITREQ.value", false
Adroit.SetTag "SOURCE1DURATIONSETPOINTREQ.value", false

'Determine the priority of the selected source by comparing source1 time to the other sources
If Source2enable = 1 Then
If Source1testtime < Source2testtime Then
  Source1Source2compare = 1
  Else
  Source1Source2compare = 0
End if
Else
  Source1Source2compare = 0
End if
If Source3enable = 1 Then
If Source1testtime < Source3testtime Then
  Source1Source3compare = 1
  Else
```

```
  Source1Source3compare = 0
End if
Else
  Source1Source3compare = 0
End if
If Source4enable = 1 Then
If Source1testtime < Source4testtime Then
  Source1Source4compare = 1
  Else
  Source1Source4compare = 0
End if
Else
  Source1Source4compare = 0
End if
If Source5enable = 1 Then
If Source1testtime < Source5testtime Then
  Source1Source5compare = 1
  Else
  Source1Source5compare = 0
End if
Else
  Source1Source5compare = 0
End if
If Source6enable = 1 Then
```

```
If Source1testtime < Source6testtime Then
   Source1Source6compare = 1
   Else
   Source1Source6compare = 0
End if
Else
   Source1Source6compare = 0
End if

Source1priority = Sourcetotalenable - (Source1Source2compare + Source1Source3compare +
      Source1Source4compare + Source1Source5compare + Source1Source6compare)

'Select Source1 as per the priority determined
If Source1priority = 1 and Priority1timeselected = true Then
   Source1priority1selected = true
   Adroit.SetTag "SOURCE1SELECTED.value", Source1priority1selected
   Adroit.SetTag "SOURCE2SELECTED.value", false
   Adroit.SetTag "SOURCE3SELECTED.value", false
   Adroit.SetTag "SOURCE4SELECTED.value", false
   Adroit.SetTag "SOURCE5SELECTED.value", false
   Adroit.SetTag "SOURCE6SELECTED.value", false
   Elseif Source1priority = 2 and Priority2timeselected = true Then
   Source1priority2selected = true
   Adroit.SetTag "SOURCE1SELECTED.value", Source1priority2selected
```

```
Adroit.SetTag "SOURCE2SELECTED.value", false
Adroit.SetTag "SOURCE3SELECTED.value", false
Adroit.SetTag "SOURCE4SELECTED.value", false
Adroit.SetTag "SOURCE5SELECTED.value", false
Adroit.SetTag "SOURCE6SELECTED.value", false
Elseif Source1priority = 3 and Priority3timeselected = true Then
Source1priority3selected = true
Adroit.SetTag "SOURCE1SELECTED.value", Source1priority3selected
Adroit.SetTag "SOURCE2SELECTED.value", false
Adroit.SetTag "SOURCE3SELECTED.value", false
Adroit.SetTag "SOURCE4SELECTED.value", false
Adroit.SetTag "SOURCE5SELECTED.value", false
Adroit.SetTag "SOURCE6SELECTED.value", false
Elseif Source1priority = 4 and Priority4timeselected = true Then
Source1priority4selected = true
Adroit.SetTag "SOURCE1SELECTED.value", Source1priority4selected
Adroit.SetTag "SOURCE2SELECTED.value", false
Adroit.SetTag "SOURCE3SELECTED.value", false
Adroit.SetTag "SOURCE4SELECTED.value", false
Adroit.SetTag "SOURCE5SELECTED.value", false
Adroit.SetTag "SOURCE6SELECTED.value", false
Elseif Source1priority = 5 and Priority5timeselected = true Then
Source1priority5selected = true
Adroit.SetTag "SOURCE1SELECTED.value", Source1priority5selected
```

```
   Adroit.SetTag "SOURCE2SELECTED.value", false
   Adroit.SetTag "SOURCE3SELECTED.value", false
   Adroit.SetTag "SOURCE4SELECTED.value", false
   Adroit.SetTag "SOURCE5SELECTED.value", false
   Adroit.SetTag "SOURCE6SELECTED.value", false
   Else
   If Source1active = false Then
   Adroit.SetTag "SOURCE1SELECTED.value", 0
End if
End if

'Source1 time placed into a priority time slot
If Source1priority = 1 Then
   Priority1time = Source1testtime
End if
If Source1priority = 2 Then
   Priority2time = Source1testtime
End if
If Source1priority = 3 Then
   Priority3time = Source1testtime
End if
If Source1priority = 4 Then
   Priority4time = Source1testtime
End if
```

```
If Source1priority = 5 Then
   Priority5time = Source1testtime
End if
If Source1priority = 6 Then
   Priority6time = Source1testtime
End if

'Source1 active condition
If Actualtime =  Source1testtime Then
   Source1active = true
   Adroit.SetTag "SOURCE1ACTIVE.value", Source1active
   Adroit.SetTag "SOURCE1SELECTED.value",Source1active
End if

'Source1 duration condition (measure samples from source)
If Source1active = true and Sourcepositionactualatsetpoint = true Then
     Do While Source1durationsetpointretrieve > (Source1durationactual + #00:00:01#) 'take the
       actual duartion 1 sec back
     Source1durationactual = Source1durationactual + Source1durationincrement
     Adroit.SetTag "SOURCE1DURATIONACTUAL.value" , Source1durationactual
     Adroit.Wait 1000 'delay the loop by 1 second
     Loop
     Source1actualdurationatsetpoint = true
End If
```

```
'Source1 start time minus offset time
If Source1active = true and Sourcepositionactualatsetpoint = true and
    Source1actualdurationatsetpoint = true Then
  Do While Source1starttimeminusoffsettime = false
  Source1starttimeoffsettimedifference = Source1starttimeretrieve - Source1offsettimeretrieve
  Source1starttimeoffsettimecompare1 = DateAdd ("s", DateDiff
    ("s",Source1offsettimeretrieve,Source1starttimeretrieve), "00:00:00")
  Source1starttimeoffsettimecompare2 = DateDiff
    ("s",Source1offsettimeretrieve,Source1starttimeretrieve)
  If Source1starttimeoffsettimecompare2 < 0 Then
  Source1starttimeoffsettimecompare3 = true
  End if
  If Source1starttimeoffsettimecompare3 = true Then
  Source1starttime = FormatDateTime (Source1starttimeoffsettimecompare1, 3)
  Adroit.SetTag "SOURCE1STARTTIME.value" , Source1starttime
  Else
  Source1starttime = FormatDateTime (Source1starttimeoffsettimedifference, 3)
  Adroit.SetTag "SOURCE1STARTTIME.value" , Source1starttime
  End if
  Source1starttimeminusoffsettime = true
  If Source1starttimeminusoffsettime = true Then
  Exit Do
  End If
```

```
  Loop
End if

'Position scan option for Source1 starting from 20deg to -20deg at 1deg/day
If Source1active = true and Sourcepositionactualatsetpoint = true and
    Source1actualdurationatsetpoint = true and Source1starttimeminusoffsettime = true Then
Do While Source1postionscansetpointminusone = false
  Source1positionscandaycount = Source1positionscandaycount + 1
  Source1positionscansetpoint = Source1positionsetpointhighlimit -
    Source1positionscandaycount
  Adroit.SetTag "SOURCE1POSITIONSETPOINT.value" , Source1positionscansetpoint
  Adroit.SetTag "SOURCE1POSITIONSPTHLIMIT.value" , Source1positionscansetpoint
  If Source1positionscansetpoint < Source1positionsetpointlowlimit Then
  Adroit.SetTag "SOURCE1ENABLE.value", 0
  Adroit.SetTag "SOURCE1POSITIONSPTHLIMIT.value" , Source1positionsetpointhighlimit
  End if
  Source1postionscansetpointminusone = true
  If Source1postionscansetpointminusone = true Then
  Exit Do
  End If
  Loop
End if

'Source 1 variable declaration reset
```

```
If Source1active = true and Sourcepositionactualatsetpoint = true and
    Source1actualdurationatsetpoint = true and Source1starttimeminusoffsettime = true Then
  Adroit.SetTag "SOURCE1SELECTED.value", false
  Source1active = false
  Adroit.SetTag "SOURCE1ACTIVE.value", Source1active
  Source1durationactual = TimeSerial (00,00,00)
  Adroit.SetTag "SOURCE1DURATIONACTUAL.value" , Source1durationactual
  Sourcepositionactualatsetpoint = false
  Source1actualdurationatsetpoint = false
  Source1starttimeminusoffsettime = false
  Source1starttimeoffsettimecompare3 = false
  Source1postionscansetpointminusone = false
  Source1sequencecompleted = true
  Source2sequencecompleted = false
  Source3sequencecompleted = false
  Source4sequencecompleted = false
  Source5sequencecompleted = false
  Source6sequencecompleted = false
End if

'Source1 disabled from here
Else
Adroit.SetTag "SOURCE1SELECTED.value", false
Source1active = false
```

```
Adroit.SetTag "SOURCE1ACTIVE.value", Source1active
Source1positionscandaycount = 0 'Reset the position day count to zero
Source1durationactual = TimeSerial (00,00,00)
Adroit.SetTag "SOURCE1DURATIONACTUAL.value" , Source1durationactual
Source1actualdurationatsetpoint = false
Source1starttimeminusoffsettime = false
Source1starttimeoffsettimecompare3 = false
Source1sequencecompleted = false

'Source1 data input
If Source1namerequest = true Then
  Source1name = (InputBox("Enter source1name","","",Inputboxxposition,Inputboxyposition))
  Adroit.SetTag "SOURCE1NAME.value" , Source1name
  Source1namerequestreset = 0
  Adroit.SetTag "SOURCE1NAMEREQ.value", Source1namerequestreset
End if
If Source1offsettimerequest = true Then
  Source1offsettime = TimeValue (InputBox("Enter source1 offset time
      (hh:mm:ss)","","",Inputboxxposition,Inputboxyposition))
  Adroit.SetTag "SOURCE1OFFSETTIME.value" , Source1offsettime
  Source1offsettimerequestreset = 0
  Adroit.SetTag "SOURCE1OFFSETTIMEREQ.value", Source1offsettimerequestreset
End if
If Source1starttimerequest = true Then
```

```
  Source1starttime = TimeValue (InputBox("Enter source1 start time
    (hh:mm:ss)","","",Inputboxxposition,Inputboxyposition))
  Adroit.SetTag "SOURCE1STARTTIME.value" , Source1starttime
  Source1starttimerequestreset = 0
  Adroit.SetTag "SOURCE1STARTTIMEREQ.value", Source1starttimerequestreset
End if

If Source1positionsetpointlowlimitrequest = true Then
  Source1positionsetpointlowlimit = Round (CDbl (InputBox("Enter source1 position setpoint
    low limit (+20deg to -20deg and less than the high
    limit)","","",Inputboxxposition,Inputboxyposition)), Roundtodecimalplace)
  If Source1positionsetpointlowlimit > Dishmaxposition or Source1positionsetpointlowlimit <
    Dishminposition Then
  MsgBox ("Invalid Entry - Angle Between +20deg to -20deg")
  Elseif Source1positionsetpointlowlimit > Source1positionsetpointhighlimit Then
  MsgBox ("Invalid Entry - Angle must be less than the high limt")
  Else
  Adroit.SetTag "SOURCE1POSITIONSPTLLIMIT.value" , Source1positionsetpointlowlimit
  Source1positionsetpointlowlimitrequest = 0
  Adroit.SetTag "SOURCE1POSITIONSPTLLIMITREQ.value",
    Source1positionsetpointlowlimitrequest
  End if
End if
```

```
If Source1positionsetpointhighlimitrequest = true Then
  Source1positionsetpointhighlimit = Round (CDbl (InputBox("Enter source1 position setpoint
    high limit (+20deg to -20deg and more than the low
    limit)","","",Inputboxxposition,Inputboxyposition)), Roundtodecimalplace)
  If Source1positionsetpointhighlimit > Dishmaxposition or Source1positionsetpointhighlimit <
    Dishminposition Then
  MsgBox ("Invalid Entry - Angle Between +20deg to -20deg")
  Elseif Source1positionsetpointhighlimit < Source1positionsetpointlowlimit Then
  MsgBox ("Invalid Entry - Angle must be more than the low limt")
  Else
  Adroit.SetTag "SOURCE1POSITIONSPTHLIMIT.value" , Source1positionsetpointhighlimit
  Adroit.SetTag "SOURCE1POSITIONSETPOINT.value" , Source1positionsetpointhighlimit
  Source1positionsetpointhighlimitrequest = 0
  Adroit.SetTag          "SOURCE1POSITIONSPTHLIMITREQ.value",
    Source1positionsetpointhighlimitrequest
  End if
End if

If Source1durationsetpointrequest = true Then
  Source1durationsetpoint = TimeValue (InputBox("Enter source1 duration setpoint
    (hh:mm:ss)","","",Inputboxxposition,Inputboxyposition))
  Adroit.SetTag "SOURCE1DURATIONSETPOINT.value" , Source1durationsetpoint
  Source1durationsetpointrequestreset = 0
```

```
  Adroit.SetTag          "SOURCE1DURATIONSETPOINTREQ.value",
      Source1durationsetpointrequestreset
End if
End if

'Source2 Enable
If Source2enable = 1 Then
'Source2 data request reset
Adroit.SetTag "SOURCE2NAMEREQ.value", false
Adroit.SetTag "SOURCE2OFFSETTIMEREQ.value", false
Adroit.SetTag "SOURCE2STARTTIMEREQ.value", false
Adroit.SetTag "SOURCE2POSITIONSETPOINTREQ.value", false
Adroit.SetTag "SOURCE2DURATIONSETPOINTREQ.value", false

'Determine the priority of the selected source by comparing source2 time to the other sources
If Source1enable = 1 Then
If Source2testtime < Source1testtime Then
  Source2Source1compare = 1
  Else
  Source2Source1compare = 0
End if
Else
  Source2Source1compare = 0
End if
```

```
If Source3enable = 1 Then
If Source2testtime < Source3testtime Then
  Source2Source3compare = 1
  Else
  Source2Source3compare = 0
End if
Else
  Source2Source3compare = 0
End if
If Source4enable = 1 Then
If Source2testtime < Source4testtime Then
  Source2Source4compare = 1
  Else
  Source2Source4compare = 0
End if
Else
  Source2Source4compare = 0
End if
If Source5enable = 1 Then
If Source2testtime < Source5testtime Then
  Source2Source5compare = 1
  Else
  Source2Source5compare = 0
End if
```

```
Else
  Source2Source5compare = 0
End if
If Source6enable = 1 Then
If Source2testtime < Source6testtime Then
  Source2Source6compare = 1
  Else
  Source2Source6compare = 0
End if
Else
  Source2Source6compare = 0
End if

Source2priority = Sourcetotalenable - (Source2Source1compare + Source2Source3compare +
      Source2Source4compare + Source2Source5compare + Source2Source6compare)

'Select Source2 as per the priority determined
If Source2priority = 1 and Priority1timeselected = true Then
  Source2priority1selected = true
  Adroit.SetTag "SOURCE2SELECTED.value", Source2priority1selected
  Adroit.SetTag "SOURCE1SELECTED.value", false
  Adroit.SetTag "SOURCE3SELECTED.value", false
  Adroit.SetTag "SOURCE4SELECTED.value", false
  Adroit.SetTag "SOURCE5SELECTED.value", false
```

```
Adroit.SetTag "SOURCE6SELECTED.value", false
Elseif Source2priority = 2 and Priority2timeselected = true Then
Source2priority2selected = true
Adroit.SetTag "SOURCE2SELECTED.value", Source2priority2selected
Adroit.SetTag "SOURCE1SELECTED.value", false
Adroit.SetTag "SOURCE3SELECTED.value", false
Adroit.SetTag "SOURCE4SELECTED.value", false
Adroit.SetTag "SOURCE5SELECTED.value", false
Adroit.SetTag "SOURCE6SELECTED.value", false
Elseif Source2priority = 3 and Priority3timeselected = true Then
Source2priority3selected = true
Adroit.SetTag "SOURCE2SELECTED.value", Source2priority3selected
Adroit.SetTag "SOURCE1SELECTED.value", false
Adroit.SetTag "SOURCE3SELECTED.value", false
Adroit.SetTag "SOURCE4SELECTED.value", false
Adroit.SetTag "SOURCE5SELECTED.value", false
Adroit.SetTag "SOURCE6SELECTED.value", false
Elseif Source2priority = 4 and Priority4timeselected = true Then
Source2priority4selected = true
Adroit.SetTag "SOURCE2SELECTED.value", Source2priority4selected
Adroit.SetTag "SOURCE1SELECTED.value", false
Adroit.SetTag "SOURCE3SELECTED.value", false
Adroit.SetTag "SOURCE4SELECTED.value", false
Adroit.SetTag "SOURCE5SELECTED.value", false
```

```
   Adroit.SetTag "SOURCE6SELECTED.value", false
   Elseif Source2priority = 5 and Priority5timeselected = true Then
   Source2priority5selected = true
   Adroit.SetTag "SOURCE2SELECTED.value", Source2priority5selected
   Adroit.SetTag "SOURCE1SELECTED.value", false
   Adroit.SetTag "SOURCE3SELECTED.value", false
   Adroit.SetTag "SOURCE4SELECTED.value", false
   Adroit.SetTag "SOURCE5SELECTED.value", false
   Adroit.SetTag "SOURCE6SELECTED.value", false
   Else
   If Source2active = false Then
   Adroit.SetTag "SOURCE2SELECTED.value", 0
End if
End if

'Source2 time placed into a priority time slot
If Source2priority = 1 Then
   Priority1time = Source2testtime
End if
If Source2priority = 2 Then
   Priority2time = Source2testtime
End if
If Source2priority = 3 Then
   Priority3time = Source2testtime
```

```
End if
If Source2priority = 4 Then
   Priority4time = Source2testtime
End if
If Source2priority = 5 Then
   Priority5time = Source2testtime
End if
If Source2priority = 6 Then
   Priority6time = Source2testtime
End if

'Source2 active condition
If Actualtime =  Source2testtime Then
   Source2active = true
   Adroit.SetTag "SOURCE2ACTIVE.value", Source2active
   Adroit.SetTag "SOURCE2SELECTED.value",Source2active
End if

'Source2 duration condition (measure samples from source)
If Source2active = true and Sourcepositionactualatsetpoint = true Then
     Do While Source2durationsetpointretrieve > (Source2durationactual + #00:00:01#) 'take the
       actual duartion 1 sec back
     Source2durationactual = Source2durationactual + Source2durationincrement
     Adroit.SetTag "SOURCE2DURATIONACTUAL.value" , Source2durationactual
```

```
    Adroit.Wait 1000 'delay the loop by 1 second
    Loop
    Source2actualdurationatsetpoint = true
End If

'Source2 start time minus offset time
If Source2active = true and Sourcepositionactualatsetpoint = true and
      Source2actualdurationatsetpoint = true Then
  Do While Source2starttimeminusoffsettime = false
  Source2starttimeoffsettimedifference = Source2starttimeretrieve - Source2offsettimeretrieve
  Source2starttimeoffsettimecompare1 = DateAdd ("s", DateDiff
      ("s",Source2offsettimeretrieve,Source2starttimeretrieve), "00:00:00")
  Source2starttimeoffsettimecompare2 = DateDiff
      ("s",Source2offsettimeretrieve,Source2starttimeretrieve)
  If Source2starttimeoffsettimecompare2 < 0 Then
  Source2starttimeoffsettimecompare3 = true
  End if
  If Source2starttimeoffsettimecompare3 = true Then
  Source2starttime = FormatDateTime (Source2starttimeoffsettimecompare1, 3)
  Adroit.SetTag "SOURCE2STARTTIME.value" , Source2starttime
  Else
  Source2starttime = FormatDateTime (Source2starttimeoffsettimedifference, 3)
  Adroit.SetTag "SOURCE2STARTTIME.value" , Source2starttime
  End if
```

```
   Source2starttimeminusoffsettime = true
   If Source2starttimeminusoffsettime = true Then
   Exit Do
   End If
   Loop
End if

If Source2active = true and Sourcepositionactualatsetpoint = true and
      Source2actualdurationatsetpoint = true and Source2starttimeminusoffsettime = true Then
   Adroit.SetTag "SOURCE2SELECTED.value", false
   Source2active = false
   Adroit.SetTag "SOURCE2ACTIVE.value", Source2active
   Source2durationactual = TimeSerial (00,00,00)
   Adroit.SetTag "SOURCE2DURATIONACTUAL.value" , Source2durationactual
   Sourcepositionactualatsetpoint = false
   Source2actualdurationatsetpoint = false
   Source2starttimeminusoffsettime = false
   Source2starttimeoffsettimecompare3 = false
   Source2sequencecompleted = true
   Source1sequencecompleted = false
   Source3sequencecompleted = false
   Source4sequencecompleted = false
   Source5sequencecompleted = false
   Source6sequencecompleted = false
```

```
End if


'Source2 disabled from here
Else
Adroit.SetTag "SOURCE2SELECTED.value", false
Source2active = false
Adroit.SetTag "SOURCE2ACTIVE.value", Source2active
Source2durationactual = TimeSerial (00,00,00)
Adroit.SetTag "SOURCE2DURATIONACTUAL.value" , Source2durationactual
Source2actualdurationatsetpoint = false
Source2starttimeminusoffsettime = false
Source2starttimeoffsettimecompare3 = false
Source2sequencecompleted = false


'Source2 Data
If Source2namerequest = true Then
  Source2name = (InputBox("Enter source2name","","",Inputboxxposition,Inputboxyposition))
  Adroit.SetTag "SOURCE2NAME.value" , Source2name
  Source2namerequestreset = 0
  Adroit.SetTag "SOURCE2NAMEREQ.value", Source2namerequestreset
End if
If Source2offsettimerequest = true Then
  Source2offsettime = TimeValue (InputBox("Enter source2 offset time
      (hh:mm:ss)","","",Inputboxxposition,Inputboxyposition))
```

```
  Adroit.SetTag "SOURCE2OFFSETTIME.value" , Source2offsettime
  Source2offsettimerequestreset = 0
  Adroit.SetTag "SOURCE2OFFSETTIMEREQ.value", Source2offsettimerequestreset
End if
If Source2starttimerequest = true Then
  Source2starttime = TimeValue (InputBox("Enter source2 start time
    (hh:mm:ss)","","",Inputboxxposition,Inputboxyposition))
  Adroit.SetTag "SOURCE2STARTTIME.value" , Source2starttime
  Source2starttimerequestreset = 0
  Adroit.SetTag "SOURCE2STARTTIMEREQ.value", Source2starttimerequestreset
End if
If Source2positionsetpointrequest = true Then
  Source2positionsetpoint = Round (CDbl (InputBox("Enter source2 position setpoint (+20deg to
    -20deg)","","",Inputboxxposition,Inputboxyposition)), Roundtodecimalplace)
  If Source2positionsetpoint > Dishmaxposition or Source2positionsetpoint < Dishminposition
    Then
  MsgBox ("Invalid Entry - Angle Between +20deg to -20deg")
  Else
  Adroit.SetTag "SOURCE2POSITIONSETPOINT.value" , Source2positionsetpoint
  Source2positionsetpointrequestreset = 0
  Adroit.SetTag "SOURCE2POSITIONSETPOINTREQ.value",
    Source2positionsetpointrequestreset
  End if
End if
```

```
If Source2durationsetpointrequest = true Then
  Source2durationsetpoint = TimeValue (InputBox("Enter source2 duration setpoint
    (hh:mm:ss)","","",Inputboxxposition,Inputboxyposition))
  Adroit.SetTag "SOURCE2DURATIONSETPOINT.value" , Source2durationsetpoint
  Source2durationsetpointrequestreset = 0
  Adroit.SetTag "SOURCE2DURATIONSETPOINTREQ.value",
    Source2durationsetpointrequestreset
End if
End if

'Source3 Enable
If Source3enable = 1 Then
'Source3 data request reset
Adroit.SetTag "SOURCE3NAMEREQ.value", false
Adroit.SetTag "SOURCE3OFFSETTIMEREQ.value", false
Adroit.SetTag "SOURCE3STARTTIMEREQ.value", false
Adroit.SetTag "SOURCE3POSITIONSETPOINTREQ.value", false
Adroit.SetTag "SOURCE3DURATIONSETPOINTREQ.value", false

'Determine the priority of the selected source by comparing source3 time to the other sources
If Source1enable = 1 Then
If Source3testtime < Source1testtime Then
  Source3Source1compare = 1
  Else
```

```
  Source3Source1compare = 0
End if
Else
  Source3Source1compare = 0
End if
If Source2enable = 1 Then
If Source3testtime < Source2testtime Then
  Source3Source2compare = 1
  Else
  Source3Source2compare = 0
End if
Else
  Source3Source2compare = 0
End if
If Source4enable = 1 Then
If Source3testtime < Source4testtime Then
  Source3Source4compare = 1
  Else
  Source3Source4compare = 0
End if
Else
  Source3Source4compare = 0
End if
If Source5enable = 1 Then
```

```
If Source3testtime < Source5testtime Then
  Source3Source5compare = 1
  Else
  Source3Source5compare = 0
End if
Else
  Source3Source5compare = 0
End if
If Source6enable = 1 Then
If Source3testtime < Source6testtime Then
  Source3Source6compare = 1
  Else
  Source3Source6compare = 0
End if
Else
  Source3Source6compare = 0
End if

Source3priority = Sourcetotalenable - (Source3Source1compare + Source3Source2compare +
      Source3Source4compare + Source3Source5compare + Source3Source6compare)

'Select Source3 as per the priority determined
If Source3priority = 1 and Priority1timeselected = true Then
  Source3priority1selected = true
```

```
Adroit.SetTag "SOURCE3SELECTED.value", Source3priority1selected
Adroit.SetTag "SOURCE1SELECTED.value", false
Adroit.SetTag "SOURCE2SELECTED.value", false
Adroit.SetTag "SOURCE4SELECTED.value", false
Adroit.SetTag "SOURCE5SELECTED.value", false
Adroit.SetTag "SOURCE6SELECTED.value", false
Elseif Source3priority = 2 and Priority2timeselected = true Then
Source3priority2selected = true
Adroit.SetTag "SOURCE3SELECTED.value", Source3priority2selected
Adroit.SetTag "SOURCE1SELECTED.value", false
Adroit.SetTag "SOURCE2SELECTED.value", false
Adroit.SetTag "SOURCE4SELECTED.value", false
Adroit.SetTag "SOURCE5SELECTED.value", false
Adroit.SetTag "SOURCE6SELECTED.value", false
Elseif Source3priority = 3 and Priority3timeselected = true Then
Source3priority3selected = true
Adroit.SetTag "SOURCE3SELECTED.value", Source3priority3selected
Adroit.SetTag "SOURCE1SELECTED.value", false
Adroit.SetTag "SOURCE2SELECTED.value", false
Adroit.SetTag "SOURCE4SELECTED.value", false
Adroit.SetTag "SOURCE5SELECTED.value", false
Adroit.SetTag "SOURCE6SELECTED.value", false
Elseif Source3priority = 4 and Priority4timeselected = true Then
Source3priority4selected = true
```

```
   Adroit.SetTag "SOURCE3SELECTED.value", Source3priority4selected
   Adroit.SetTag "SOURCE1SELECTED.value", false
   Adroit.SetTag "SOURCE2SELECTED.value", false
   Adroit.SetTag "SOURCE4SELECTED.value", false
   Adroit.SetTag "SOURCE5SELECTED.value", false
   Adroit.SetTag "SOURCE6SELECTED.value", false
   Elseif Source3priority = 5 and Priority5timeselected = true Then
   Source3priority5selected = true
   Adroit.SetTag "SOURCE3SELECTED.value", Source3priority5selected
   Adroit.SetTag "SOURCE1SELECTED.value", false
   Adroit.SetTag "SOURCE2SELECTED.value", false
   Adroit.SetTag "SOURCE4SELECTED.value", false
   Adroit.SetTag "SOURCE5SELECTED.value", false
   Adroit.SetTag "SOURCE6SELECTED.value", false
   Else
   If Source3active = false Then
   Adroit.SetTag "SOURCE3SELECTED.value", 0
End if
End if

'Source3 time placed into a priority time slot
If Source3priority = 1 Then
   Priority1time = Source3testtime
End if
```

```
If Source3priority = 2 Then
   Priority2time = Source3testtime
End if
If Source3priority = 3 Then
   Priority3time = Source3testtime
End if
If Source3priority = 4 Then
   Priority4time = Source3testtime
End if
If Source3priority = 5 Then
   Priority5time = Source3testtime
End if
If Source3priority = 6 Then
   Priority6time = Source3testtime
End if

'Source3 active condition
If Actualtime =  Source3testtime Then
   Source3active = true
   Adroit.SetTag "SOURCE3ACTIVE.value", Source3active
   Adroit.SetTag "SOURCE3SELECTED.value",Source3active
End if

'Source3 duration condition (measure samples from source)
```

```
If Source3active = true and Sourcepositionactualatsetpoint = true Then
    Do While Source3durationsetpointretrieve > (Source3durationactual + #00:00:01#) 'take the
      actual duartion 1 sec back
    Source3durationactual = Source3durationactual + Source3durationincrement
    Adroit.SetTag "SOURCE3DURATIONACTUAL.value" , Source3durationactual
    Adroit.Wait 1000 'delay the loop by 1 second
    Loop
    Source3actualdurationatsetpoint = true
End If

'Source3 start time minus offset time
If Source3active = true and Sourcepositionactualatsetpoint = true and
      Source3actualdurationatsetpoint = true Then
  Do While Source3starttimeminusoffsettime = false
  Source3starttimeoffsettimedifference = Source3starttimeretrieve - Source3offsettimeretrieve
  Source3starttimeoffsettimecompare1 = DateAdd ("s", DateDiff
      ("s",Source3offsettimeretrieve,Source3starttimeretrieve), "00:00:00")
  Source3starttimeoffsettimecompare2 = DateDiff
      ("s",Source3offsettimeretrieve,Source3starttimeretrieve)
  If Source3starttimeoffsettimecompare2 < 0 Then
  Source3starttimeoffsettimecompare3 = true
  End if
  If Source3starttimeoffsettimecompare3 = true Then
  Source3starttime = FormatDateTime (Source3starttimeoffsettimecompare1, 3)
```

```
    Adroit.SetTag "SOURCE3STARTTIME.value" , Source3starttime
    Else
    Source3starttime = FormatDateTime (Source3starttimeoffsettimedifference, 3)
    Adroit.SetTag "SOURCE3STARTTIME.value" , Source3starttime
    End if
    Source3starttimeminusoffsettime = true
    If Source3starttimeminusoffsettime = true Then
    Exit Do
    End If
    Loop
End if

If Source3active = true and Sourcepositionactualatsetpoint = true and
      Source3actualdurationatsetpoint = true and Source3starttimeminusoffsettime = true Then
    Adroit.SetTag "SOURCE3SELECTED.value", false
    Source3active = false
    Adroit.SetTag "SOURCE3ACTIVE.value", Source3active
    Source3durationactual = TimeSerial (00,00,00)
    Adroit.SetTag "SOURCE3DURATIONACTUAL.value" , Source3durationactual
    Sourcepositionactualatsetpoint = false
    Source3actualdurationatsetpoint = false
    Source3starttimeminusoffsettime = false
    Source3starttimeoffsettimecompare3 = false
    Source3sequencecompleted = true
```

```
   Source1sequencecompleted = false
   Source2sequencecompleted = false
   Source4sequencecompleted = false
   Source5sequencecompleted = false
   Source6sequencecompleted = false
End if

'Source3 disabled from here
Else
Adroit.SetTag "SOURCE3SELECTED.value", false
Source3active = false
Adroit.SetTag "SOURCE3ACTIVE.value", Source3active
Source3durationactual = TimeSerial (00,00,00)
Adroit.SetTag "SOURCE3DURATIONACTUAL.value" , Source3durationactual
Source3actualdurationatsetpoint = false
Source3starttimeminusoffsettime = false
Source3starttimeoffsettimecompare3 = false
Source3sequencecompleted = false

'Source3 Data
If Source3namerequest = true Then
   Source3name = (InputBox("Enter source3name","","",Inputboxxposition,Inputboxyposition))
   Adroit.SetTag "SOURCE3NAME.value" , Source3name
   Source3namerequestreset = 0
```

```
  Adroit.SetTag "SOURCE3NAMEREQ.value", Source3namerequestreset
End if
If Source3offsettimerequest = true Then
  Source3offsettime = TimeValue (InputBox("Enter source3 offset time
    (hh:mm:ss)","","",Inputboxxposition,Inputboxyposition))
  Adroit.SetTag "SOURCE3OFFSETTIME.value" , Source3offsettime
  Source3offsettimerequestreset = 0
  Adroit.SetTag "SOURCE3OFFSETTIMEREQ.value", Source3offsettimerequestreset
End if
If Source3starttimerequest = true Then
  Source3starttime = TimeValue (InputBox("Enter source3 start time
    (hh:mm:ss)","","",Inputboxxposition,Inputboxyposition))
  Adroit.SetTag "SOURCE3STARTTIME.value" , Source3starttime
  Source3starttimerequestreset = 0
  Adroit.SetTag "SOURCE3STARTTIMEREQ.value", Source3starttimerequestreset
End if
If Source3positionsetpointrequest = true Then
  Source3positionsetpoint = Round (CDbl (InputBox("Enter source3 position setpoint (+20deg to
    -20deg)","","",Inputboxxposition,Inputboxyposition)), Roundtodecimalplace)
  If Source3positionsetpoint > Dishmaxposition or Source3positionsetpoint < Dishminposition
    Then
  MsgBox ("Invalid Entry - Angle Between +20deg to -20deg")
  Else
  Adroit.SetTag "SOURCE3POSITIONSETPOINT.value" , Source3positionsetpoint
```

```
  Source3positionsetpointrequestreset = 0
  Adroit.SetTag          "SOURCE3POSITIONSETPOINTREQ.value",
      Source3positionsetpointrequestreset
  End if
End if
If Source3durationsetpointrequest = true Then
  Source3durationsetpoint = TimeValue (InputBox("Enter source3 duration setpoint
      (hh:mm:ss)","","",Inputboxxposition,Inputboxyposition))
  Adroit.SetTag "SOURCE3DURATIONSETPOINT.value" , Source3durationsetpoint
  Source3durationsetpointrequestreset = 0
  Adroit.SetTag          "SOURCE3DURATIONSETPOINTREQ.value",
      Source3durationsetpointrequestreset
End if
End if

'Source4 Enable
If Source4enable = 1 Then
'Source4 data request reset
Adroit.SetTag "SOURCE4NAMEREQ.value", false
Adroit.SetTag "SOURCE4OFFSETTIMEREQ.value", false
Adroit.SetTag "SOURCE4STARTTIMEREQ.value", false
Adroit.SetTag "SOURCE4POSITIONSETPOINTREQ.value", false
Adroit.SetTag "SOURCE4DURATIONSETPOINTREQ.value", false
```

```
'Determine the priority of the selected source by comparing source4 time to the other sources
If Source1enable = 1 Then
If Source4testtime < Source1testtime Then
   Source4Source1compare = 1
   Else
   Source4Source1compare = 0
End if
Else
   Source4Source1compare = 0
End if
If Source2enable = 1 Then
If Source4testtime < Source2testtime Then
   Source4Source2compare = 1
   Else
   Source4Source2compare = 0
End if
Else
   Source4Source2compare = 0
End if
If Source3enable = 1 Then
If Source4testtime < Source3testtime Then
   Source4Source3compare = 1
   Else
   Source4Source3compare = 0
```

```
End if
Else
  Source4Source3compare = 0
End if
If Source5enable = 1 Then
If Source4testtime < Source5testtime Then
  Source4Source5compare = 1
  Else
  Source4Source5compare = 0
End if
Else
  Source4Source5compare = 0
End if
If Source6enable = 1 Then
If Source4testtime < Source6testtime Then
  Source4Source6compare = 1
  Else
  Source4Source6compare = 0
End if
Else
  Source4Source6compare = 0
End if
```

```
Source4priority = Sourcetotalenable - (Source4Source1compare + Source4Source2compare +
    Source4Source3compare + Source4Source5compare + Source4Source6compare)

'Select Source4 as per the priority determined
If Source4priority = 1 and Priority1timeselected = true Then
  Source4priority1selected = true
  Adroit.SetTag "SOURCE4SELECTED.value", Source4priority1selected
  Adroit.SetTag "SOURCE1SELECTED.value", false
  Adroit.SetTag "SOURCE2SELECTED.value", false
  Adroit.SetTag "SOURCE3SELECTED.value", false
  Adroit.SetTag "SOURCE5SELECTED.value", false
  Adroit.SetTag "SOURCE6SELECTED.value", false
  Elseif Source4priority = 2 and Priority2timeselected = true Then
  Source4priority2selected = true
  Adroit.SetTag "SOURCE4SELECTED.value", Source4priority2selected
  Adroit.SetTag "SOURCE1SELECTED.value", false
  Adroit.SetTag "SOURCE2SELECTED.value", false
  Adroit.SetTag "SOURCE3SELECTED.value", false
  Adroit.SetTag "SOURCE5SELECTED.value", false
  Adroit.SetTag "SOURCE6SELECTED.value", false
  Elseif Source4priority = 3 and Priority3timeselected = true Then
  Source4priority3selected = true
  Adroit.SetTag "SOURCE4SELECTED.value", Source4priority3selected
  Adroit.SetTag "SOURCE1SELECTED.value", false
```

```
Adroit.SetTag "SOURCE2SELECTED.value", false
Adroit.SetTag "SOURCE3SELECTED.value", false
Adroit.SetTag "SOURCE5SELECTED.value", false
Adroit.SetTag "SOURCE6SELECTED.value", false
Elseif Source4priority = 4 and Priority4timeselected = true Then
Source4priority4selected = true
Adroit.SetTag "SOURCE4SELECTED.value", Source4priority4selected
Adroit.SetTag "SOURCE1SELECTED.value", false
Adroit.SetTag "SOURCE2SELECTED.value", false
Adroit.SetTag "SOURCE3SELECTED.value", false
Adroit.SetTag "SOURCE5SELECTED.value", false
Adroit.SetTag "SOURCE6SELECTED.value", false
Elseif Source4priority = 5 and Priority5timeselected = true Then
Source4priority5selected = true
Adroit.SetTag "SOURCE4SELECTED.value", Source4priority5selected
Adroit.SetTag "SOURCE1SELECTED.value", false
Adroit.SetTag "SOURCE2SELECTED.value", false
Adroit.SetTag "SOURCE3SELECTED.value", false
Adroit.SetTag "SOURCE5SELECTED.value", false
Adroit.SetTag "SOURCE6SELECTED.value", false
Else
If Source4active = false Then
Adroit.SetTag "SOURCE4SELECTED.value", 0
End if
```

```
End if


'Source4 time placed into a priority time slot
If Source4priority = 1 Then
   Priority1time = Source4testtime
End if
If Source4priority = 2 Then
   Priority2time = Source4testtime
End if
If Source4priority = 3 Then
   Priority3time = Source4testtime
End if
If Source4priority = 4 Then
   Priority4time = Source4testtime
End if
If Source4priority = 5 Then
   Priority5time = Source4testtime
End if
If Source4priority = 6 Then
   Priority6time = Source4testtime
End if


'Source4 active condition
If Actualtime =  Source4testtime Then
```

```
  Source4active = true
  Adroit.SetTag "SOURCE4ACTIVE.value", Source4active
  Adroit.SetTag "SOURCE4SELECTED.value",Source4active
End if

'Source4 duration condition (measure samples from source)
If Source4active = true and Sourcepositionactualatsetpoint = true Then
    Do While Source4durationsetpointretrieve > (Source4durationactual + #00:00:01#) 'take the
      actual duartion 1 sec back
    Source4durationactual = Source4durationactual + Source4durationincrement
    Adroit.SetTag "SOURCE4DURATIONACTUAL.value" , Source4durationactual
    Adroit.Wait 1000 'delay the loop by 1 second
    Loop
    Source4actualdurationatsetpoint = true
End If

'Source4 start time minus offset time
If Source4active = true and Sourcepositionactualatsetpoint = true and
      Source4actualdurationatsetpoint = true Then
  Do While Source4starttimeminusoffsettime = false
  Source4starttimeoffsettimedifference = Source4starttimeretrieve - Source4offsettimeretrieve
  Source4starttimeoffsettimecompare1 = DateAdd ("s", DateDiff
      ("s",Source4offsettimeretrieve,Source4starttimeretrieve), "00:00:00")
```

```
  Source4starttimeoffsettimecompare2 = DateDiff
     ("s",Source4offsettimeretrieve,Source4starttimeretrieve)
  If Source4starttimeoffsettimecompare2 < 0 Then
  Source4starttimeoffsettimecompare3 = true
  End if
  If Source4starttimeoffsettimecompare3 = true Then
  Source4starttime = FormatDateTime (Source4starttimeoffsettimecompare1, 3)
  Adroit.SetTag "SOURCE4STARTTIME.value" , Source4starttime
  Else
  Source4starttime = FormatDateTime (Source4starttimeoffsettimedifference, 3)
  Adroit.SetTag "SOURCE4STARTTIME.value" , Source4starttime
  End if
  Source4starttimeminusoffsettime = true
  If Source4starttimeminusoffsettime = true Then
  Exit Do
  End If
  Loop
End if

If Source4active = true and Sourcepositionactualatsetpoint = true and
     Source4actualdurationatsetpoint = true and Source4starttimeminusoffsettime = true Then
  Adroit.SetTag "SOURCE4SELECTED.value", false
  Source4active = false
  Adroit.SetTag "SOURCE4ACTIVE.value", Source4active
```

```
  Source4durationactual = TimeSerial (00,00,00)
  Adroit.SetTag "SOURCE4DURATIONACTUAL.value" , Source4durationactual
  Sourcepositionactualatsetpoint = false
  Source4actualdurationatsetpoint = false
  Source4starttimeminusoffsettime = false
  Source4starttimeoffsettimecompare3 = false
  Source4sequencecompleted = true
  Source1sequencecompleted = false
  Source2sequencecompleted = false
  Source3sequencecompleted = false
  Source5sequencecompleted = false
  Source6sequencecompleted = false
End if

'Source4 disabled from here
Else
Adroit.SetTag "SOURCE4SELECTED.value", false
Source4active = false
Adroit.SetTag "SOURCE4ACTIVE.value", Source4active
Source4durationactual = TimeSerial (00,00,00)
Adroit.SetTag "SOURCE4DURATIONACTUAL.value" , Source4durationactual
Source4actualdurationatsetpoint = false
Source4starttimeminusoffsettime = false
Source4starttimeoffsettimecompare3 = false
```

```
Source4sequencecompleted = false


'Source4 Data
If Source4namerequest = true Then
  Source4name = (InputBox("Enter source4name","","",Inputboxxposition,Inputboxyposition))
  Adroit.SetTag "SOURCE4NAME.value" , Source4name
  Source4namerequestreset = 0
  Adroit.SetTag "SOURCE4NAMEREQ.value", Source4namerequestreset
End if
If Source4offsettimerequest = true Then
  Source4offsettime = TimeValue (InputBox("Enter source4 offset time
      (hh:mm:ss)","","",Inputboxxposition,Inputboxyposition))
  Adroit.SetTag "SOURCE4OFFSETTIME.value" , Source4offsettime
  Source4offsettimerequestreset = 0
  Adroit.SetTag "SOURCE4OFFSETTIMEREQ.value", Source4offsettimerequestreset
End if
If Source4starttimerequest = true Then
  Source4starttime = TimeValue (InputBox("Enter source4 start time
      (hh:mm:ss)","","",Inputboxxposition,Inputboxyposition))
  Adroit.SetTag "SOURCE4STARTTIME.value" , Source4starttime
  Source4starttimerequestreset = 0
  Adroit.SetTag "SOURCE4STARTTIMEREQ.value", Source4starttimerequestreset
End if
If Source4positionsetpointrequest = true Then
```

```
  Source4positionsetpoint = Round (CDbl (InputBox("Enter source4 position setpoint (+20deg to
      -20deg)","","",Inputboxxposition,Inputboxyposition)), Roundtodecimalplace)
  If Source4positionsetpoint > Dishmaxposition or Source4positionsetpoint < Dishminposition
      Then
  MsgBox ("Invalid Entry - Angle Between +20deg to -20deg")
  Else
  Adroit.SetTag "SOURCE4POSITIONSETPOINT.value" , Source4positionsetpoint
  Source4positionsetpointrequestreset = 0
  Adroit.SetTag          "SOURCE4POSITIONSETPOINTREQ.value",
      Source4positionsetpointrequestreset
  End if
End if
If Source4durationsetpointrequest = true Then
  Source4durationsetpoint = TimeValue (InputBox("Enter source4 duration setpoint
      (hh:mm:ss)","","",Inputboxxposition,Inputboxyposition))
  Adroit.SetTag "SOURCE4DURATIONSETPOINT.value" , Source4durationsetpoint
  Source4durationsetpointrequestreset = 0
  Adroit.SetTag          "SOURCE4DURATIONSETPOINTREQ.value",
      Source4durationsetpointrequestreset
End if
End if

'Source5 Enable
If Source5enable = 1 Then
```

```
'Source5 data request reset
Adroit.SetTag "SOURCE5NAMEREQ.value", false
Adroit.SetTag "SOURCE5OFFSETTIMEREQ.value", false
Adroit.SetTag "SOURCE5STARTTIMEREQ.value", false
Adroit.SetTag "SOURCE5POSITIONSETPOINTREQ.value", false
Adroit.SetTag "SOURCE5DURATIONSETPOINTREQ.value", false

'Determine the priority of the selected source by comparing source5 time to the other sources
If Source1enable = 1 Then
If Source5testtime < Source1testtime Then
  Source5Source1compare = 1
  Else
  Source5Source1compare = 0
End if
Else
  Source5Source1compare = 0
End if
If Source2enable = 1 Then
If Source5testtime < Source2testtime Then
  Source5Source2compare = 1
  Else
  Source5Source2compare = 0
End if
Else
```

```
  Source5Source2compare = 0
End if
If Source3enable = 1 Then
If Source5testtime < Source3testtime Then
  Source5Source3compare = 1
  Else
  Source5Source3compare = 0
End if
Else
  Source5Source3compare = 0
End if
If Source4enable = 1 Then
If Source5testtime < Source4testtime Then
  Source5Source4compare = 1
  Else
  Source5Source4compare = 0
End if
Else
  Source5Source4compare = 0
End if
If Source6enable = 1 Then
If Source5testtime < Source6testtime Then
  Source5Source6compare = 1
  Else
```

```
   Source5Source6compare = 0
End if
Else
   Source5Source6compare = 0
End if

Source5priority = Sourcetotalenable - (Source5Source1compare + Source5Source2compare +
      Source5Source3compare + Source5Source4compare + Source5Source6compare)

'Select Source5 as per the priority determined
If Source5priority = 1 and Priority1timeselected = true Then
   Source5priority1selected = true
   Adroit.SetTag "SOURCE5SELECTED.value", Source5priority1selected
   Adroit.SetTag "SOURCE1SELECTED.value", false
   Adroit.SetTag "SOURCE2SELECTED.value", false
   Adroit.SetTag "SOURCE3SELECTED.value", false
   Adroit.SetTag "SOURCE4SELECTED.value", false
   Adroit.SetTag "SOURCE6SELECTED.value", false
   Elseif Source5priority = 2 and Priority2timeselected = true Then
   Source5priority2selected = true
   Adroit.SetTag "SOURCE5SELECTED.value", Source5priority2selected
   Adroit.SetTag "SOURCE1SELECTED.value", false
   Adroit.SetTag "SOURCE2SELECTED.value", false
   Adroit.SetTag "SOURCE3SELECTED.value", false
```

```
Adroit.SetTag "SOURCE4SELECTED.value", false
Adroit.SetTag "SOURCE6SELECTED.value", false
Elseif Source5priority = 3 and Priority3timeselected = true Then
Source5priority3selected = true
Adroit.SetTag "SOURCE5SELECTED.value", Source5priority3selected
Adroit.SetTag "SOURCE1SELECTED.value", false
Adroit.SetTag "SOURCE2SELECTED.value", false
Adroit.SetTag "SOURCE3SELECTED.value", false
Adroit.SetTag "SOURCE4SELECTED.value", false
Adroit.SetTag "SOURCE6SELECTED.value", false
Elseif Source5priority = 4 and Priority4timeselected = true Then
Source5priority4selected = true
Adroit.SetTag "SOURCE5SELECTED.value", Source5priority4selected
Adroit.SetTag "SOURCE1SELECTED.value", false
Adroit.SetTag "SOURCE2SELECTED.value", false
Adroit.SetTag "SOURCE3SELECTED.value", false
Adroit.SetTag "SOURCE4SELECTED.value", false
Adroit.SetTag "SOURCE6SELECTED.value", false
Elseif Source5priority = 5 and Priority5timeselected = true Then
Source5priority5selected = true
Adroit.SetTag "SOURCE5SELECTED.value", Source5priority5selected
Adroit.SetTag "SOURCE1SELECTED.value", false
Adroit.SetTag "SOURCE2SELECTED.value", false
Adroit.SetTag "SOURCE3SELECTED.value", false
```

```
   Adroit.SetTag "SOURCE4SELECTED.value", false
   Adroit.SetTag "SOURCE6SELECTED.value", false
   Else
   If Source5active = false Then
   Adroit.SetTag "SOURCE5SELECTED.value", 0
End if
End if

'Source5 time placed into a priority time slot
If Source5priority = 1 Then
   Priority1time = Source5testtime
End if
If Source5priority = 2 Then
   Priority2time = Source5testtime
End if
If Source5priority = 3 Then
   Priority3time = Source5testtime
End if
If Source5priority = 4 Then
   Priority4time = Source5testtime
End if
If Source5priority = 5 Then
   Priority5time = Source5testtime
End if
```

```
If Source5priority = 6 Then
   Priority6time = Source5testtime
End if

'Source5 active condition
If Actualtime =  Source5testtime Then
   Source5active = true
   Adroit.SetTag "SOURCE5ACTIVE.value", Source5active
   Adroit.SetTag "SOURCE5SELECTED.value",Source5active
End if

'Source5 duration condition (measure samples from source)
If Source5active = true and Sourcepositionactualatsetpoint = true Then
     Do While Source5durationsetpointretrieve > (Source5durationactual + #00:00:01#) 'take the
       actual duartion 1 sec back
     Source5durationactual = Source5durationactual + Source5durationincrement
     Adroit.SetTag "SOURCE5DURATIONACTUAL.value" , Source5durationactual
     Adroit.Wait 1000 'delay the loop by 1 second
     Loop
     Source5actualdurationatsetpoint = true
End If

'Source5 start time minus offset time
```

```
If Source5active = true and Sourcepositionactualatsetpoint = true and
    Source5actualdurationatsetpoint = true Then
  Do While Source5starttimeminusoffsettime = false
  Source5starttimeoffsettimedifference = Source5starttimeretrieve - Source5offsettimeretrieve
  Source5starttimeoffsettimecompare1 = DateAdd ("s", DateDiff
    ("s",Source5offsettimeretrieve,Source5starttimeretrieve), "00:00:00")
  Source5starttimeoffsettimecompare2 = DateDiff
    ("s",Source5offsettimeretrieve,Source5starttimeretrieve)
  If Source5starttimeoffsettimecompare2 < 0 Then
  Source5starttimeoffsettimecompare3 = true
  End if
  If Source5starttimeoffsettimecompare3 = true Then
  Source5starttime = FormatDateTime (Source5starttimeoffsettimecompare1, 3)
  Adroit.SetTag "SOURCE5STARTTIME.value" , Source5starttime
  Else
  Source5starttime = FormatDateTime (Source5starttimeoffsettimedifference, 3)
  Adroit.SetTag "SOURCE5STARTTIME.value" , Source5starttime
  End if
  Source5starttimeminusoffsettime = true
  If Source5starttimeminusoffsettime = true Then
  Exit Do
  End If
  Loop
End if
```

```
If Source5active = true and Sourcepositionactualatsetpoint = true and
    Source5actualdurationatsetpoint = true and Source5starttimeminusoffsettime = true Then
  Adroit.SetTag "SOURCE5SELECTED.value", false
  Source5active = false
  Adroit.SetTag "SOURCE5ACTIVE.value", Source5active
  Source5durationactual = TimeSerial (00,00,00)
  Adroit.SetTag "SOURCE5DURATIONACTUAL.value" , Source5durationactual
  Sourcepositionactualatsetpoint = false
  Source5actualdurationatsetpoint = false
  Source5starttimeminusoffsettime = false
  Source5starttimeoffsettimecompare3 = false
  Source5sequencecompleted = true
  Source1sequencecompleted = false
  Source2sequencecompleted = false
  Source3sequencecompleted = false
  Source4sequencecompleted = false
  Source6sequencecompleted = false
End if

'Source5 disabled from here
Else
Adroit.SetTag "SOURCE5SELECTED.value", false
Source5active = false
```

```
Adroit.SetTag "SOURCE5ACTIVE.value", Source5active
Source5durationactual = TimeSerial (00,00,00)
Adroit.SetTag "SOURCE5DURATIONACTUAL.value" , Source5durationactual
Source5actualdurationatsetpoint = false
Source5starttimeminusoffsettime = false
Source5starttimeoffsettimecompare3 = false
Source5sequencecompleted = false

'Source5 Data
If Source5namerequest = true Then
  Source5name = (InputBox("Enter source5name","","",Inputboxxposition,Inputboxyposition))
  Adroit.SetTag "SOURCE5NAME.value" , Source5name
  Source5namerequestreset = 0
  Adroit.SetTag "SOURCE5NAMEREQ.value", Source5namerequestreset
End if
If Source5offsettimerequest = true Then
  Source5offsettime = TimeValue (InputBox("Enter source5 offset time
      (hh:mm:ss)","","",Inputboxxposition,Inputboxyposition))
  Adroit.SetTag "SOURCE5OFFSETTIME.value" , Source5offsettime
  Source5offsettimerequestreset = 0
  Adroit.SetTag "SOURCE5OFFSETTIMEREQ.value", Source5offsettimerequestreset
End if
If Source5starttimerequest = true Then
```

```
  Source5starttime = TimeValue (InputBox("Enter source5 start time
      (hh:mm:ss)","","",Inputboxxposition,Inputboxyposition))
  Adroit.SetTag "SOURCE5STARTTIME.value" , Source5starttime
  Source5starttimerequestreset = 0
  Adroit.SetTag "SOURCE5STARTTIMEREQ.value", Source5starttimerequestreset
End if
If Source5positionsetpointrequest = true Then
  Source5positionsetpoint = Round (CDbl (InputBox("Enter source5 position setpoint (+20deg to
      -20deg)","","",Inputboxxposition,Inputboxyposition)), Roundtodecimalplace)
  If Source5positionsetpoint > Dishmaxposition or Source5positionsetpoint < Dishminposition
      Then
  MsgBox ("Invalid Entry - Angle Between +20deg to -20deg")
  Else
  Adroit.SetTag "SOURCE5POSITIONSETPOINT.value" , Source5positionsetpoint
  Source5positionsetpointrequestreset = 0
  Adroit.SetTag "SOURCE5POSITIONSETPOINTREQ.value",
      Source5positionsetpointrequestreset
  End if
End if
If Source5durationsetpointrequest = true Then
  Source5durationsetpoint = TimeValue (InputBox("Enter source5 duration setpoint
      (hh:mm:ss)","","",Inputboxxposition,Inputboxyposition))
  Adroit.SetTag "SOURCE5DURATIONSETPOINT.value" , Source5durationsetpoint
  Source5durationsetpointrequestreset = 0
```

```
  Adroit.SetTag          "SOURCE5DURATIONSETPOINTREQ.value",
      Source5durationsetpointrequestreset
End if
End if

'Source6 Enable
If Source6enable = 1 Then
'Source5 data request reset
Adroit.SetTag "SOURCE6NAMEREQ.value", false
Adroit.SetTag "SOURCE6OFFSETTIMEREQ.value", false
Adroit.SetTag "SOURCE6STARTTIMEREQ.value", false
Adroit.SetTag "SOURCE6POSITIONSETPOINTREQ.value", false
Adroit.SetTag "SOURCE6DURATIONSETPOINTREQ.value", false

'Determine the priority of the selected source by comparing source6 time to the other sources
If Source1enable = 1 Then
If Source6testtime < Source1testtime Then
  Source6Source1compare = 1
  Else
  Source6Source1compare = 0
End if
Else
  Source6Source1compare = 0
End if
```

```
If Source2enable = 1 Then
If Source6testtime < Source2testtime Then
  Source6Source2compare = 1
  Else
  Source6Source2compare = 0
End if
Else
  Source6Source2compare = 0
End if
If Source3enable = 1 Then
If Source6testtime < Source3testtime Then
  Source6Source3compare = 1
  Else
  Source6Source3compare = 0
End if
Else
  Source6Source3compare = 0
End if
If Source4enable = 1 Then
If Source6testtime < Source4testtime Then
  Source6Source4compare = 1
  Else
  Source6Source4compare = 0
End if
```

```
Else
  Source6Source4compare = 0
End if
If Source5enable = 1 Then
If Source6testtime < Source5testtime Then
  Source6Source5compare = 1
  Else
  Source6Source5compare = 0
End if
Else
  Source6Source5compare = 0
End if

Source6priority = Sourcetotalenable - (Source6Source1compare + Source6Source2compare +
      Source6Source3compare + Source6Source4compare + Source6Source5compare)

'Select Source6 as per the priority determined
If Source6priority = 1 and Priority1timeselected = true Then
  Source6priority1selected = true
  Adroit.SetTag "SOURCE6SELECTED.value", Source6priority1selected
  Adroit.SetTag "SOURCE1SELECTED.value", false
  Adroit.SetTag "SOURCE2SELECTED.value", false
  Adroit.SetTag "SOURCE3SELECTED.value", false
  Adroit.SetTag "SOURCE4SELECTED.value", false
```

```
Adroit.SetTag "SOURCE5SELECTED.value", false
Elseif Source6priority = 2 and Priority2timeselected = true Then
Source6priority2selected = true
Adroit.SetTag "SOURCE6SELECTED.value", Source6priority2selected
Adroit.SetTag "SOURCE1SELECTED.value", false
Adroit.SetTag "SOURCE2SELECTED.value", false
Adroit.SetTag "SOURCE3SELECTED.value", false
Adroit.SetTag "SOURCE4SELECTED.value", false
Adroit.SetTag "SOURCE5SELECTED.value", false
Elseif Source6priority = 3 and Priority3timeselected = true Then
Source6priority3selected = true
Adroit.SetTag "SOURCE6SELECTED.value", Source6priority3selected
Adroit.SetTag "SOURCE1SELECTED.value", false
Adroit.SetTag "SOURCE2SELECTED.value", false
Adroit.SetTag "SOURCE3SELECTED.value", false
Adroit.SetTag "SOURCE4SELECTED.value", false
Adroit.SetTag "SOURCE5SELECTED.value", false
Elseif Source6priority = 4 and Priority4timeselected = true Then
Source6priority4selected = true
Adroit.SetTag "SOURCE6SELECTED.value", Source6priority4selected
Adroit.SetTag "SOURCE1SELECTED.value", false
Adroit.SetTag "SOURCE2SELECTED.value", false
Adroit.SetTag "SOURCE3SELECTED.value", false
Adroit.SetTag "SOURCE4SELECTED.value", false
```

```
Adroit.SetTag "SOURCE5SELECTED.value", false
Elseif Source6priority = 5 and Priority5timeselected = true Then
Source6priority5selected = true
Adroit.SetTag "SOURCE6SELECTED.value", Source6priority5selected
Adroit.SetTag "SOURCE1SELECTED.value", false
Adroit.SetTag "SOURCE2SELECTED.value", false
Adroit.SetTag "SOURCE3SELECTED.value", false
Adroit.SetTag "SOURCE4SELECTED.value", false
Adroit.SetTag "SOURCE5SELECTED.value", false
Elseif Source6priority = 6 and Priority6timeselected = true Then
Source6priority6selected = true
Adroit.SetTag "SOURCE6SELECTED.value", Source6priority6selected
Adroit.SetTag "SOURCE1SELECTED.value", false
Adroit.SetTag "SOURCE2SELECTED.value", false
Adroit.SetTag "SOURCE3SELECTED.value", false
Adroit.SetTag "SOURCE4SELECTED.value", false
Adroit.SetTag "SOURCE5SELECTED.value", false
Else
If Source6active = false Then
Adroit.SetTag "SOURCE6SELECTED.value", 0
End if
End if

'Source6 time placed into a priority time slot
```

```
If Source6priority = 1 Then
   Priority1time = Source6testtime
End if
If Source6priority = 2 Then
   Priority2time = Source6testtime
End if
If Source6priority = 3 Then
   Priority3time = Source6testtime
End if
If Source6priority = 4 Then
   Priority4time = Source6testtime
End if
If Source6priority = 5 Then
   Priority5time = Source6testtime
End if
If Source6priority = 6 Then
   Priority6time = Source6testtime
End if

'Source6 active condition
If Actualtime =  Source6testtime Then
   Source6active = true
   Adroit.SetTag "SOURCE6ACTIVE.value", Source6active
   Adroit.SetTag "SOURCE6SELECTED.value",Source6active
```

```
End if


'Source6 duration condition (measure samples from source)
If Source6active = true and Sourcepositionactualatsetpoint = true Then
    Do While Source6durationsetpointretrieve > (Source6durationactual + #00:00:01#) 'take the
      actual duartion 1 sec back
    Source6durationactual = Source6durationactual + Source6durationincrement
    Adroit.SetTag "SOURCE6DURATIONACTUAL.value" , Source6durationactual
    Adroit.Wait 1000 'delay the loop by 1 second
    Loop
    Source6actualdurationatsetpoint = true
End If


'Source6 start time minus offset time
If Source6active = true and Sourcepositionactualatsetpoint = true and
      Source6actualdurationatsetpoint = true Then
  Do While Source6starttimeminusoffsettime = false
  Source6starttimeoffsettimedifference = Source6starttimeretrieve - Source6offsettimeretrieve
  Source6starttimeoffsettimecompare1 = DateAdd ("s", DateDiff
      ("s",Source6offsettimeretrieve,Source6starttimeretrieve), "00:00:00")
  Source6starttimeoffsettimecompare2 = DateDiff
      ("s",Source6offsettimeretrieve,Source6starttimeretrieve)
  If Source6starttimeoffsettimecompare2 < 0 Then
  Source6starttimeoffsettimecompare3 = true
```

```
    End if
    If Source6starttimeoffsettimecompare3 = true Then
    Source6starttime = FormatDateTime (Source6starttimeoffsettimecompare1, 3)
    Adroit.SetTag "SOURCE6STARTTIME.value" , Source6starttime
    Else
    Source6starttime = FormatDateTime (Source6starttimeoffsettimedifference, 3)
    Adroit.SetTag "SOURCE6STARTTIME.value" , Source6starttime
    End if
    Source6starttimeminusoffsettime = true
    If Source6starttimeminusoffsettime = true Then
    Exit Do
    End If
    Loop
End if

If Source6active = true and Sourcepositionactualatsetpoint = true and
    Source6actualdurationatsetpoint = true and Source6starttimeminusoffsettime = true Then
    Adroit.SetTag "SOURCE6SELECTED.value", false
    Source6active = false
    Adroit.SetTag "SOURCE6ACTIVE.value", Source6active
    Source6durationactual = TimeSerial (00,00,00)
    Adroit.SetTag "SOURCE6DURATIONACTUAL.value" , Source6durationactual
    Sourcepositionactualatsetpoint = false
    Source6actualdurationatsetpoint = false
```

```
    Source6starttimeminusoffsettime = false
    Source6starttimeoffsettimecompare3 = false
    Source6sequencecompleted = true
    Source1sequencecompleted = false
    Source2sequencecompleted = false
    Source3sequencecompleted = false
    Source4sequencecompleted = false
    Source5sequencecompleted = false
End if

'Source6 disabled from here
Else
Adroit.SetTag "SOURCE6SELECTED.value", false
Source6active = false
Adroit.SetTag "SOURCE6ACTIVE.value", Source6active
Source6durationactual = TimeSerial (00,00,00)
Adroit.SetTag "SOURCE6DURATIONACTUAL.value" , Source6durationactual
Source6actualdurationatsetpoint = false
Source6starttimeminusoffsettime = false
Source6starttimeoffsettimecompare3 = false
Source6sequencecompleted = false

'Source6 Data
If Source6namerequest = true Then
```

```
  Source6name = (InputBox("Enter source6name","","",Inputboxxposition,Inputboxyposition))
  Adroit.SetTag "SOURCE6NAME.value" , Source6name
  Source6namerequestreset = 0
  Adroit.SetTag "SOURCE6NAMEREQ.value", Source6namerequestreset
End if
If Source6offsettimerequest = true Then
  Source6offsettime = TimeValue (InputBox("Enter source6 offset time
    (hh:mm:ss)","","",Inputboxxposition,Inputboxyposition))
  Adroit.SetTag "SOURCE6OFFSETTIME.value" , Source6offsettime
  Source6offsettimerequestreset = 0
  Adroit.SetTag "SOURCE6OFFSETTIMEREQ.value", Source6offsettimerequestreset
End if
If Source6starttimerequest = true Then
  Source6starttime = TimeValue (InputBox("Enter source6 start time
    (hh:mm:ss)","","",Inputboxxposition,Inputboxyposition))
  Adroit.SetTag "SOURCE6STARTTIME.value" , Source6starttime
  Source6starttimerequestreset = 0
  Adroit.SetTag "SOURCE6STARTTIMEREQ.value", Source6starttimerequestreset
End if
If Source6positionsetpointrequest = true Then
  Source6positionsetpoint = Round (CDbl (InputBox("Enter source6 position setpoint (+20deg to
    -20deg)","","",Inputboxxposition,Inputboxyposition)), Roundtodecimalplace)
  If Source6positionsetpoint > Dishmaxposition or Source6positionsetpoint < Dishminposition
    Then
```

```
  MsgBox ("Invalid Entry - Angle Between +20deg to -20deg")
  Else
  Adroit.SetTag "SOURCE6POSITIONSETPOINT.value" , Source6positionsetpoint
  Source6positionsetpointrequestreset = 0
  Adroit.SetTag          "SOURCE6POSITIONSETPOINTREQ.value",
      Source6positionsetpointrequestreset
  End if
End if
If Source6durationsetpointrequest = true Then
  Source6durationsetpoint = TimeValue (InputBox("Enter source6 duration setpoint
      (hh:mm:ss)","","",Inputboxxposition,Inputboxyposition))
  Adroit.SetTag "SOURCE6DURATIONSETPOINT.value" , Source6durationsetpoint
  Source6durationsetpointrequestreset = 0
  Adroit.SetTag          "SOURCE6DURATIONSETPOINTREQ.value",
      Source6durationsetpointrequestreset
End if
End if

'reset and hide enable buttons
If Source1enable = 1 Then
  Adroit.Wait 100 'delay the sequence so that the reset takes place and is not skipped
  Adroit.SetTag "SOURCE2ENABLE.value", 0 'disable Source2
  Adroit.SetTag "SOURCE3ENABLE.value", 0 'disable Source3
  Adroit.SetTag "SOURCE4ENABLE.value", 0 'disable Source4
```

```
   Adroit.SetTag "SOURCE5ENABLE.value", 0 'disable Source5
   Adroit.SetTag "SOURCE6ENABLE.value", 0 'disable Source6
   Adroit.SetTag "SOURCE2ENABLEBUTTONHIDE.value", 1 'hide Source2 enable button
   Adroit.SetTag "SOURCE3ENABLEBUTTONHIDE.value", 1 'hide Source3 enable button
   Adroit.SetTag "SOURCE4ENABLEBUTTONHIDE.value", 1 'hide Source4 enable button
   Adroit.SetTag "SOURCE5ENABLEBUTTONHIDE.value", 1 'hide Source5 enable button
   Adroit.SetTag "SOURCE6ENABLEBUTTONHIDE.value", 1 'hide Source6 enable button
End if

If Source2enable = 1 or Source3enable = 1 or Source4enable = 1 or Source5enable = 1 or
      Source6enable = 1 Then
   Adroit.Wait 100 'delay the sequence so that the reset takes place and is not skipped
   Adroit.SetTag "SOURCE1ENABLE.value", 0 'disable Source1
   Adroit.SetTag "SOURCE1ENABLEBUTTONHIDE.value", 1 'hide Source1 enable button
End if

'show enable buttons
If Source1enable = 0 Then
   Adroit.SetTag "SOURCE2ENABLEBUTTONHIDE.value", 0 'show Source2 enable button
   Adroit.SetTag "SOURCE3ENABLEBUTTONHIDE.value", 0 'show Source3 enable button
   Adroit.SetTag "SOURCE4ENABLEBUTTONHIDE.value", 0 'show Source4 enable button
   Adroit.SetTag "SOURCE5ENABLEBUTTONHIDE.value", 0 'show Source5 enable button
   Adroit.SetTag "SOURCE6ENABLEBUTTONHIDE.value", 0 'show Source6 enable button
End if
```

```
If Source2enable = 0 and Source3enable = 0 and Source4enable = 0 and Source5enable = 0 and
      Source6enable = 0 Then
  Adroit.SetTag "SOURCE1ENABLEBUTTONHIDE.value", 0 'show Source1 enable button
End if

'Priority time selection
If Actualtime <= Priority1time Then
  Priority1timeselected = true
  Else
  Priority1timeselected = false
End if
If Actualtime > Priority1time and Actualtime <= Priority2time Then
  Priority2timeselected = true
  Else
  Priority2timeselected = false
End if
If Actualtime > Priority2time and Actualtime <= Priority3time Then
  Priority3timeselected = true
  Else
  Priority3timeselected = false
End if
If Actualtime > Priority3time and Actualtime <= Priority4time Then
  Priority4timeselected = true
```

```
  Else
  Priority4timeselected = false
End if
If Actualtime > Priority4time and Actualtime <= Priority5time Then
  Priority5timeselected = true
  Else
  Priority5timeselected = false
End if
If Actualtime > Priority5time and Actualtime <= Priority6time Then
  Priority6timeselected = true
  Else
  Priority6timeselected = false
End if

'Move dish actual position to setpoint position
If Source1active = true Then
  Adroit.SetTag "SOURCE2SELECTED.value", false
  Source2active = false
  Adroit.SetTag "SOURCE2ACTIVE.value", Source2active
  Adroit.SetTag "SOURCE3SELECTED.value", false
  Source3active = false
  Adroit.SetTag "SOURCE3ACTIVE.value", Source3active
  Adroit.SetTag "SOURCE4SELECTED.value", false
  Source4active = false
```

```
Adroit.SetTag "SOURCE4ACTIVE.value", Source4active
Adroit.SetTag "SOURCE5SELECTED.value", false
Source5active = false
Adroit.SetTag "SOURCE5ACTIVE.value", Source5active
Adroit.SetTag "SOURCE6SELECTED.value", false
Source6active = false
Adroit.SetTag "SOURCE6ACTIVE.value", Source6active
Source1positionsetpointtoPLC = CInt (Adroit.GetTag
    ("SOURCE1POSITIONSETPOINT.value"))
SourceautopositionsetpointtoPLC = Source1positionsetpointtoPLC
Elseif Source2active = true Then
Adroit.SetTag "SOURCE1SELECTED.value", false
Source1active = false
Adroit.SetTag "SOURCE1ACTIVE.value", Source1active
Adroit.SetTag "SOURCE3SELECTED.value", false
Source3active = false
Adroit.SetTag "SOURCE3ACTIVE.value", Source3active
Adroit.SetTag "SOURCE4SELECTED.value", false
Source4active = false
Adroit.SetTag "SOURCE4ACTIVE.value", Source4active
Adroit.SetTag "SOURCE5SELECTED.value", false
Source5active = false
Adroit.SetTag "SOURCE5ACTIVE.value", Source5active
Adroit.SetTag "SOURCE6SELECTED.value", false
```

```
Source6active = false
Adroit.SetTag "SOURCE6ACTIVE.value", Source6active
Source2positionsetpointtoPLC = CInt (Adroit.GetTag
    ("SOURCE2POSITIONSETPOINT.value"))
SourceautopositionsetpointtoPLC = Source2positionsetpointtoPLC
Elseif Source3active = true Then
Adroit.SetTag "SOURCE1SELECTED.value", false
Source1active = false
Adroit.SetTag "SOURCE1ACTIVE.value", Source1active
Adroit.SetTag "SOURCE2SELECTED.value", false
Source2active = false
Adroit.SetTag "SOURCE2ACTIVE.value", Source2active
Adroit.SetTag "SOURCE4SELECTED.value", false
Source4active = false
Adroit.SetTag "SOURCE4ACTIVE.value", Source4active
Adroit.SetTag "SOURCE5SELECTED.value", false
Source5active = false
Adroit.SetTag "SOURCE5ACTIVE.value", Source5active
Adroit.SetTag "SOURCE6SELECTED.value", false
Source6active = false
Adroit.SetTag "SOURCE6ACTIVE.value", Source6active
Source3positionsetpointtoPLC = CInt (Adroit.GetTag
    ("SOURCE3POSITIONSETPOINT.value"))
SourceautopositionsetpointtoPLC = Source3positionsetpointtoPLC
```

```
Elseif Source4active = true Then
Adroit.SetTag "SOURCE1SELECTED.value", false
Source1active = false
Adroit.SetTag "SOURCE1ACTIVE.value", Source1active
Adroit.SetTag "SOURCE2SELECTED.value", false
Source2active = false
Adroit.SetTag "SOURCE2ACTIVE.value", Source2active
Adroit.SetTag "SOURCE3SELECTED.value", false
Source3active = false
Adroit.SetTag "SOURCE3ACTIVE.value", Source3active
Adroit.SetTag "SOURCE5SELECTED.value", false
Source5active = false
Adroit.SetTag "SOURCE5ACTIVE.value", Source5active
Adroit.SetTag "SOURCE6SELECTED.value", false
Source6active = false
Adroit.SetTag "SOURCE6ACTIVE.value", Source6active
Source4positionsetpointtoPLC = CInt (Adroit.GetTag
    ("SOURCE4POSITIONSETPOINT.value"))
SourceautopositionsetpointtoPLC = Source4positionsetpointtoPLC
Elseif Source5active = true Then
Adroit.SetTag "SOURCE1SELECTED.value", false
Source1active = false
Adroit.SetTag "SOURCE1ACTIVE.value", Source1active
Adroit.SetTag "SOURCE2SELECTED.value", false
```

```
Source2active = false
Adroit.SetTag "SOURCE2ACTIVE.value", Source2active
Adroit.SetTag "SOURCE3SELECTED.value", false
Source3active = false
Adroit.SetTag "SOURCE3ACTIVE.value", Source3active
Adroit.SetTag "SOURCE4SELECTED.value", false
Source4active = false
Adroit.SetTag "SOURCE4ACTIVE.value", Source4active
Adroit.SetTag "SOURCE6SELECTED.value", false
Source6active = false
Adroit.SetTag "SOURCE6ACTIVE.value", Source6active
Source5positionsetpointtoPLC = CInt (Adroit.GetTag
    ("SOURCE5POSITIONSETPOINT.value"))
SourceautopositionsetpointtoPLC = Source5positionsetpointtoPLC
Elseif Source6active = true Then
Adroit.SetTag "SOURCE1SELECTED.value", false
Source1active = false
Adroit.SetTag "SOURCE1ACTIVE.value", Source1active
Adroit.SetTag "SOURCE2SELECTED.value", false
Source2active = false
Adroit.SetTag "SOURCE2ACTIVE.value", Source2active
Adroit.SetTag "SOURCE3SELECTED.value", false
Source3active = false
Adroit.SetTag "SOURCE3ACTIVE.value", Source3active
```

```
  Adroit.SetTag "SOURCE4SELECTED.value", false
  Source4active = false
  Adroit.SetTag "SOURCE4ACTIVE.value", Source4active
  Adroit.SetTag "SOURCE5SELECTED.value", false
  Source5active = false
  Adroit.SetTag "SOURCE5ACTIVE.value", Source5active
  Source6positionsetpointtoPLC = CInt (Adroit.GetTag
     ("SOURCE6POSITIONSETPOINT.value"))
  SourceautopositionsetpointtoPLC = Source6positionsetpointtoPLC
End if

If (Source1active = true or Source2active = true or Source3active = true or Source4active = true
     or Source5active = true or Source6active = true) and SourceautopositionsetpointtoPLC >
     SourcepositionactualfromPLC and not SourceautopositionsetpointtoPLC > Dishmaxposition
     Then 'Source postion actual below setpoint
   Do While SourceautopositionsetpointtoPLC > SourcepositionactualfromPLC
   SourcepositionactualfromPLC = Adroit.GetTag ("PV_ANGLE_FL.value")
   Adroit.SetTag "SCADA_ANGLE_SP_EXP.v01" , SourceautopositionsetpointtoPLC
   If (SourcepositionactualfromPLC + Toleranceplus) >= SourceautopositionsetpointtoPLC
     Then
   Sourcepositionactualatsetpoint = true
   Exit Do
   End If
   Loop
```

```
End If

If (Source1active = true or Source2active = true or Source3active = true or Source4active = true
    or Source5active = true or Source6active = true) and SourceautopositionsetpointtoPLC <
    SourcepositionactualfromPLC and not SourceautopositionsetpointtoPLC < Dishminposition
    Then 'Source postion actual above setpoint
  Do While SourceautopositionsetpointtoPLC < SourcepositionactualfromPLC
  SourcepositionactualfromPLC = Adroit.GetTag ("PV_ANGLE_FL.value")
  Adroit.SetTag "SCADA_ANGLE_SP_EXP.v01" , SourceautopositionsetpointtoPLC
  If (SourcepositionactualfromPLC - Toleranceminus) <= SourceautopositionsetpointtoPLC
    Then
  Sourcepositionactualatsetpoint = true
  Exit Do
  End If
  Loop
End If

If (Source1active = true or Source2active = true or Source3active = true or Source4active = true
    or Source5active = true or Source6active = true) and (((SourcepositionactualfromPLC +
    Toleranceplus) >= SourceautopositionsetpointtoPLC) or ((SourcepositionactualfromPLC -
    Toleranceminus) <= SourceautopositionsetpointtoPLC)) and not
    SourceautopositionsetpointtoPLC > Dishmaxposition and not
    SourceautopositionsetpointtoPLC < Dishminposition Then
  Sourcepositionactualatsetpoint = true
```

```
End if

'Automatic disabled
   Sourceautomanual = Adroit.GetTag ("SOURCEAUTOMANUAL.value")
   If Sourceautomanual = false Then
   Exit Do
   End If
   Loop

End Sub

'Main software end
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
```

The SCADA automanual script has comments describing certain visual basic code functionality. The following describes that not described in the visual basic script:

- Dim Sourceautomanual

Dim, is used to declare a variable explicitly.

- Adroit.SetTag "SOURCE1ENABLE.value", 0

The SetTag function assigns a value to a variable. The SOURCE1ENABLE.value is how a tag in the PLC is referenced.

- Source1durationactual = TimeSerial (00,00,00)

The TimeSerial function returns a variant with the time for a specific hour, minute and second.

- Sourceautomanual = Adroit.GetTag ("SOURCEAUTOMANUAL.value")

The GetTag function gets the value of a tag.

```
If Sourcemanualpositionsetpointtoplc <= Dishmaxposition or Sourcemanualpositionsetpointtoplc >= Dishminposition Then
   Sourcemanualpositionsetpointrequest = Adroit.GetTag ("SOURCEMANUALPOSITIONSPTREQ.value")

If Sourcemanualpositionsetpointrequest = true Then
   Sourcemanualpositionsetpointtoplc = Round (CDbl (InputBox("Enter dish manual position setpoint (+20deg to -20deg)","","",Inputboxxposition,Inputboxypos
   Sourcemanualpositionsetpointcheck = IsNumeric (Sourcemanualpositionsetpointtoplc)
   Adroit.SetTag "SCADA_ANGLE_SP_EXP.v01" , Sourcemanualpositionsetpointtoplc

If Sourcemanualpositionsetpointtoplc > Dishmaxposition or Sourcemanualpositionsetpointtoplc < Dishminposition Then
   MsgBox ("Invalid Entry - Angle Between +20deg to -20deg")
End If
```

If the manual setpoint box is selected, an input window appears with a comment. This allows for a manual position setpoint to be entered. The setpoint entered must be a number and within the range specified. If the setpoint is outside this range, then an error message is displayed.

```
If Sourcemanualpositionsetpointtoplc > Sourcepositionactualfromplc and not Sourcemanualpositionsetpointtoplc > Dishmaxposition Then
      Do While Sourcemanualpositionsetpointtoplc > Sourcepositionactualfromplc
      Sourcepositionactualfromplc = Adroit.GetTag ("PV_ANGLE_FL.value")
      Sourcemanualpositionsetpointrequestreset = 0
      Adroit.SetTag "SOURCEMANUALPOSITIONSPTREQ.value" , Sourcemanualpositionsetpointrequestreset
      Sourcemanualpositionsetpointrequest = Adroit.GetTag ("SOURCEMANUALPOSITIONSPTREQ.value")
      If (Sourcepositionactualfromplc + Toleranceplus) >= Sourcemanualpositionsetpointtoplc Then
      Exit Do
      End If
      Loop
End If

If Sourcemanualpositionsetpointtoplc < Sourcepositionactualfromplc and not Sourcemanualpositionsetpointtoplc < Dishminposition Then
      Do While Sourcemanualpositionsetpointtoplc < Sourcepositionactualfromplc
      Sourcepositionactualfromplc = Adroit.GetTag ("PV_ANGLE_FL.value")
      Sourcemanualpositionsetpointrequestreset = 0
      Adroit.SetTag "SOURCEMANUALPOSITIONSPTREQ.value" , Sourcemanualpositionsetpointrequestreset
      Sourcemanualpositionsetpointrequest = Adroit.GetTag ("SOURCEMANUALPOSITIONSPTREQ.value")
      If (Sourcepositionactualfromplc - Toleranceminus) <= Sourcemanualpositionsetpointtoplc Then
      Exit Do
      End If
      Loop
End If
```

The direction in which the antenna must move is determined by this code. If the actual position is above or below the setpoint, the antenna will move to the desired setpoint within a specific tolerance as defined.

- Actualtime = CDate(Time)

The CDate function converts the variable from a string to a time value.

- Source1enable = Abs (CInt (Adroit.GetTag ("SOURCE1ENABLE.value")))

The CInt function converts a value into an integer. The Abs function returns an absolute value of a number.

- Source1offsettimeretrieve=TimeValue(Adroit.GetTag("SOURCE1OFFSETTIME.value")
)

The TimeValue function converts a string to a time.

```
If Source1priority = 1 and Priority1timeselected = true Then
   Source1priority1selected = true
   Adroit.SetTag "SOURCE1SELECTED.value", Source1priority1selected
   Adroit.SetTag "SOURCE2SELECTED.value", false
   Adroit.SetTag "SOURCE3SELECTED.value", false
   Adroit.SetTag "SOURCE4SELECTED.value", false
   Adroit.SetTag "SOURCE5SELECTED.value", false
   Adroit.SetTag "SOURCE6SELECTED.value", false
```

If a source has been selected as priority and is displayed as selected, the other sources selections are forced off.

```
If Source1active = true and Sourcepositionactualatsetpoint = true Then
     Do While Source1durationsetpointretrieve > (Source1durationactual + #00:00:01#) 'take the actual duartion 1 sec back
     Source1durationactual = Source1durationactual + Source1durationincrement
     Adroit.SetTag "SOURCE1DURATIONACTUAL.value" , Source1durationactual
     Adroit.Wait 1000 'delay the loop by 1 second
     Loop
     Source1actualdurationatsetpoint = true
End If
```

Once the antenna reaches its desired position setpoint, the duration to measure the source is started. The duration actual will increment from zero, in increments of one second, until it reaches the duration setpoint. Once the duration actual equals the duration setpoint, the loop is exited.

- Source1starttimeoffsettimecompare1=DateAdd("s",DateDiff("s",Source1offsettimeretrieve,Source1starttimeretrieve), "00:00:00")

The DateAdd function returns a date to which a specified time interval has been added. The DateDiff function returns the difference between the offset time and start time in seconds..

The manual functionality of the system can be describes as follows:

To operate the system in manual, select the manual position by clicking on the selector switch on the main overview screen. The position setpoint can be changed by selecting the variable. This displays an input window where the desired setpoint can be entered. The system has been designed to only accept values between a maximum (+20°) and minimum (-20°) limit. Any values entered outside these limits or alphanumeric characters will display an invalid entry message. Once a valid setpoint value has been entered and accepted, the system displays this and moves the antenna from its current position to the new setpoint position.

The automatic functionality of the system can be describes as follows:

To operate the system in automatic, select the auto position by clicking on he selector switch on the main overview screen and navigate to the Source screen by selecting the Source option available on the menu screen. There are six sources available for selection. Source two to six can be individually selected and when selected source 1 becomes unavailable. There are five parameters that have to be entered namely; celestial source, offset time, start time, position

setpoint and duration setpoint. These parameters can only be entered when all of these sources are not selected.

- The celestial source is the name of the source. The name of the source can be changed by selecting the variable. This displays an input window were the name can be entered. The system has been designed to accept any characters. Once a name has been entered and accepted, the system will display this.

- The position setpoint is the position that the antenna needs to move to, from its current position, controlled to 0.35° tolerance. The position setpoint can be changed by selecting the variable. This displays an input window where the desired setpoint can be entered. The system has been designed to only accept values between a maximum (+20°) and minimum (-20°) limit. Any values entered outside these limits or alphanumeric characters will display an invalid entry message. Once a valid setpoint value has been entered and accepted, the system displays this.

- The start time is when the antenna will start moving from its current position to the new setpoint position. The start time can be changed by selecting the variable. This displays an input window where the desired start time can be entered. The system has been designed to only accept a value in the following format (hh:mm:ss). Once a valid time value has been entered and accepted, the system displays this.

- The duration setpoint is the time the antenna will stay in this new position to take measurements of the source. The duration setpoint can be changed by selecting the

variable. This displays an input window were the desired duration setpoint can be entered. The system has been designed to only accept a time value. Once a valid time value has been entered and accepted, the system displays this. The duration actual will start incrementing every one second. Once the duration actual is equal to the duration setpoint, the measurement period is completed. The start time will be changed backwards by the offset time and become the new start time.

- The offset time is the time that is based on the earths rotation relative to the celestial source and is used to change the start time backwards. The offset time can be changed by selecting the variable. This displays an input window where the desired offset time can be entered. The system has been designed to only accept a time value. Once a valid time value has been entered and accepted, the system displays this.

On selection of each source, a prompt is displayed to confirm the selection. When multiple sources are selected, the system is designed to determine the priority of each selected source relative to the start times where the earliest start time will be first priority. The source that has priority to start next is displayed as the selected source. This source operational sequence is activated and displayed, when the actual time equals the start time. The antenna will then start moving to the position setpoint. Once reached it will remain in this position so that the source can be measured and will then be deselected so that a next priority source can be selected.

When source 1 is selected, sources two to six become unavailable. The parameter options are similar to the other sources with the exception of the position setpoint parameter as this has a range option. By default the range has a maximum (+20°) and a minimum (-20°) setpoint value,

however this can be changed. The position setpoint can be changed by selecting the variable. This displays an input window where the desired setpoint can be entered. The system has been designed to only accept values between a maximum (+20°) and minimum (-20°) limit and the maximum value is greater than the minimum value. Any values entered outside these limits or alphanumeric characters will display an invalid entry message. Once a valid setpoint value has been entered and accepted, the system displays this. When this source is selected, a prompt is displayed to confirm the selection. On confirmation of the source selection and all other conditions as described above are met, the antenna will start moving from its actual position to the maximum (+20°) setpoint position. The antenna will move from this maximum (+20°) position to the minimum (-20°) position, with a 1° position change for every cycle. On completion of this cycle, the source is automatically de-selected.

APPENDIX D

SCADA WGP SAVE FILE CONFIGURATION FUNCTIONALITY

This option is used to save the Source parameters entered for automatic control. In the event of a power failure these parameters are not lost. This is achieved by clicking the save button on the source screen. This option is only available when automatic control is selected and none of the sources are in the enabled state. The procedure below describes how this is configured.

- Go to the Agent Configurator.
- Select Agent “Boolean”.
- Agent name “SAVEWGPFILE”.
- Select Agent “Expression”.
- Agent name: SAVE_WGP_FILE.
- Edit: value – SAVEWGPFILE.value (boolen), V01 – SOURCE1ENABLE.value, V02 – SOURCE2ENABLE.value, V03 – SOURCE3ENABLE.value, V04 - SOURCE4ENABLE.value, V05 - SOURCE5ENABLE.value, V06 – SOURCE6ENABLE.value, V07 – SOURCEAUTOMANUAL, expression – v1|v2|v3|v4|v5|v6!v7, trigger – none, display – names, evaluation – continuous.
- Create a “Button”, buttontext – (SAVE, font – bold, size – 6), behaviours – operator action (tag – systemInfo.saveNow, min tag/value – 0, max tag/value – 1, action – control action toggle, confirm – tick), visibility – (tag – SAVEWGPFILE.value: Boolean, visibility – invisible).

APPENDIX E

SCADA ANGLE CONVERTER SCRIPT

The angle converter is a tool that was developed to convert the antenna elevation angle of +70° to 110°, as an interpreted elevation angle of +20° to -20°. The elevation angle to be converted can be changed by selecting the variable. This displays an input window where the true angle can be entered. The system has been designed to only accept values between the ranges specified. Any values entered outside these limits or alphanumeric characters will display an invalid entry message. Once a valid true angle has been entered and accepted, the system displays the resultant angle.

```
Sub Main()

'Main software.....
'Converter variable declaration
Dim Converteranglesetpointrequest, Converteranglesetpoint, Converteranglesetpointrequestreset,
    Converterangleactual
Converteranglesetpointrequest = Adroit.GetTag
    ("CONVERTERANGLESETPOINTREQ.value")

If Converteranglesetpointrequest = true Then
  Converteranglesetpoint = CInt (InputBox("Enter angle to be convertered to +20deg to -
    20deg","","",7000,11000))
  Adroit.SetTag "CONVERTERANGLESETPOINT.value", Converteranglesetpoint
```

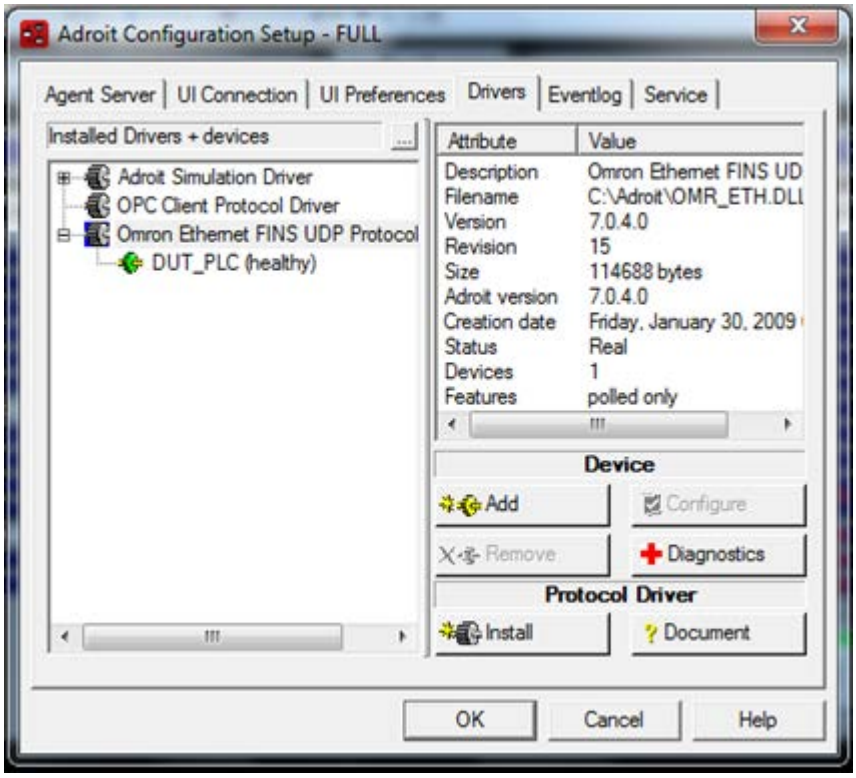
```
    Converteranglesetpointrequestreset = 0
    Adroit.SetTag          "CONVERTERANGLESETPOINTREQ.value",
        Converteranglesetpointrequestreset
If Converteranglesetpoint <= 89 and Converteranglesetpoint >= 70 Then
    Converterangleactual = (90) - Converteranglesetpoint
    Adroit.SetTag "CONVERTERANGLEACTUAL.value", Converterangleactual
    Else if Converteranglesetpoint >= -89 and Converteranglesetpoint <= -70 Then
    Converterangleactual = (-90) - Converteranglesetpoint
    Adroit.SetTag "CONVERTERANGLEACTUAL.value", Converterangleactual
    Else if Converteranglesetpoint = 90 Then
    Converterangleactual = 0
    Adroit.SetTag "CONVERTERANGLEACTUAL.value", Converterangleactual
    Else
    Adroit.SetTag "CONVERTERANGLESETPOINT.value", ""
    Adroit.SetTag "CONVERTERANGLEACTUAL.value", ""
    MsgBox ("Invalid entry - angle range: -70deg to -89deg, 90deg, 89deg to 70deg")
    End if
    End if
    End if
End if

End Sub

'Main software end
```

APPENDIX F

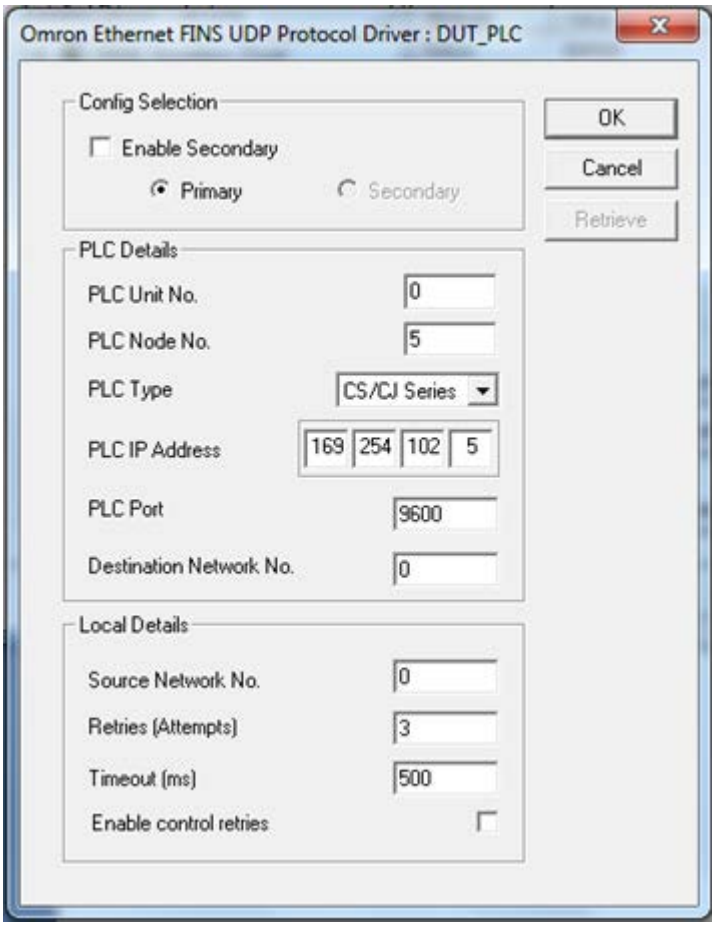
PLC AND SCADA COMMUNICATION SETUP PROCEDURE

The following procedure describes the communication setup between the SCADA and PLC.

- Install the Omron Ethernet FINS UDP protocol driver and add a device as per the following:



- Configuration: configuration selection – primary, PLC unit number – 0, PLC node number – 5, PLC type – CS/CJ series, PLC IP address – 169.254.102.5, PLC port – 9600, destination network – 0, source network – 0, retries – 3, timeouts – 500 and enable controls – not selected.



The following procedure describes an example of adding a tag.

- Open the Adroit Classic UI application and load the file.
- Go to configuration, select i.e. Boolean, input a name – OMRONINPUT0, click on the add button, select the OMRONINPUT0 tag and click on the scan button, select the Adroit server – OMRONPLC, select the start radio button, choose the address – CI00101.00, click on the scan button, this will associate the tag with the PLC.
- If you want to disassociate a tag from the PLC, select the tag and click on un-scan, this will delete the tag.

- The tag can now be used for display purposes i.e. light indication – on/off.
- Open the Adroit Setup application and select the drivers tab.
- Go to the associated PLC. The status should now read as healthy and is green.
- By clicking on the PLC and selecting the diagnostics, it brings up a diagnostics screen. Select options 1, 2 and 3.
- By selecting the parent PLC and selecting diagnostics, it will show the status of the PLC polling.

APPENDIX G

WEATHER STATION INSTALLATION AND CONFIGURATION PROCEDURE

The following describes the procedure to install and configure the weather station system software.

- Run the installer program “WeatherOS_V1.1.36_151108”.
- Install the software in the default directory “c:\Program Files\Oregon Scientific.
- Open the program and go to settings.
- Under “General”, for the Weather Station Model select: WMR 200/A
- Under “General”, for the Recording Settings enable: launch recording at windows start and do full backup daily. The Data Directory must be set to: c:\Program Files\Oregon Scientific\Weather OS.
- Under “Sensors”, leave as per default settings.
- Under “Language”, select English.
- Under “City”, select Home City.

APPENDIX H

WEATHER STATION SCRIPTS AND FUNCTIONALITY

A script has been written for each weather condition namely; the local pressure, rainfall, temperature, humidity, wind direction and wind speed.

- Weather Condition – Local Pressure

```
Sub Main()
'Main software start
excelPath = "C:\Program Files\Oregon Scientific\Weather OS\DATA\BARO\1day.csv"
Set objExcel = CreateObject("Excel.Application")
objExcel.DisplayAlerts = 0
objExcel.Workbooks.Open excelPath, False, True
Set currentWorkSheet = objExcel.ActiveWorkbook.Worksheets(1)
Set Cells = currentWorkSheet.Cells
Column = 3
Row = 1
local_pressure = Cells(Row, Column).Value
Set currentWorkSheet = Nothing
objExcel.Workbooks(1).Close
objExcel.Quit
Set objExcel = Nothing
```

```
Adroit.SetTag "WEATHERLOCALPRESSURE.value", local_pressure
End Sub
'Main software end
```

- Weather Condition – Rainfall

```
Sub Main()
'Main software start
excelPath = "C:\Program Files\Oregon Scientific\Weather OS\DATA\RAIN\1day.csv"
Set objExcel = CreateObject("Excel.Application")
objExcel.DisplayAlerts = 0
objExcel.Workbooks.Open excelPath, False, True
Set currentWorkSheet = objExcel.ActiveWorkbook.Worksheets(1)
Set Cells = currentWorkSheet.Cells
Column = 5
Row = 1
rainfall = Cells(Row, Column).Value
Set currentWorkSheet = Nothing
objExcel.Workbooks(1).Close
objExcel.Quit
Set objExcel = Nothing
Adroit.SetTag "WEATHERRAINFALL.value", rainfall
End Sub
'Main software end
```

- Weather Condition – Temperature and Humidity

```
Sub Main()
'Main software start
excelPath = "C:\Program Files\Oregon Scientific\Weather OS\DATA\TH\sensor0-1day.csv"
Set objExcel = CreateObject("Excel.Application")
objExcel.DisplayAlerts = 0
objExcel.Workbooks.Open excelPath, False, True
Set currentWorkSheet = objExcel.ActiveWorkbook.Worksheets(1)
Set Cells = currentWorkSheet.Cells
Column1 = 4
Row1 = 1
temperature = Cells(Row1, Column1).Value
Column2 = 5
Row2 = 1
humidity = Cells(Row2, Column2).Value
Set currentWorkSheet = Nothing
objExcel.Workbooks(1).Close
objExcel.Quit
Set objExcel = Nothing
Adroit.SetTag "WEATHERTEMPERATURE.value", temperature
Adroit.SetTag "WEATHERHUMIDITY.value", humidity
```

```
End Sub
'Main software end
```

- Weather Condition – Wind Direction and Wind Speed

```
Sub Main()
'Main software start
excelPath = "C:\Program Files\Oregon Scientific\Weather OS\DATA\WIND\1day.csv"
Set objExcel = CreateObject("Excel.Application")
objExcel.DisplayAlerts = 0
objExcel.Workbooks.Open excelPath, False, True
Set currentWorkSheet = objExcel.ActiveWorkbook.Worksheets(1)
Set Cells = currentWorkSheet.Cells
Column1 = 3
Row1 = 1
winddirection = Cells(Row1, Column1).Value
Column2 = 5
Row2 = 1
windspeed = Cells(Row2, Column2).Value
Set currentWorkSheet = Nothing
objExcel.Workbooks(1).Close
objExcel.Quit
Set objExcel = Nothing
```

```
Adroit.SetTag "WEATHERWINDDIRECTION.value", winddirection
Adroit.SetTag "WEATHERWINDSPEED.value", windspeed
End Sub
'Main software end
```

The weather station script specifies the directory of the comma separated value (CSV) file location. This file is then opened using the Excel application. It then points to the first worksheet in this file and the cell location as per its column and row position. The data in this cell is then transferred to the tag that is used to display the value on the SCADA mimic. The Script then closes the file and quits the Excel application. The following describes the script visual basic code per line:

The location of the file is specified as the “excelPath”.

- excelPath = "C:\Program Files\Oregon Scientific\Weather OS\DATA\BARO\1day.csv"

This binds the Excel application to the Excel object so that the file can be opened using Excel.

- Set objExcel = CreateObject("Excel.Application")

Opens the file using Excel that is stored in the “excelPath” location.

- objExcel.Workbooks.Open excelPath, False, True

Used to specify that the script must work with the first worksheet in the workbook.

- Set currentWorkSheet = objExcel.ActiveWorkbook.Worksheets(1)

Sets up the active sheets cell location in terms of column and row position.

- Set Cells = currentWorkSheet.Cells, Column = 3, Row = 1

The cell value is transferred to “local_pressure” as a string value.

- local_pressure = Cells(Row, Column).Value

Closes the active workbook i.e. Excel file.

- objExcel.Workbooks(1).Close

This statement is used to quit the Excel application.

- objExcel.Quit

Clean up of the Excel object.

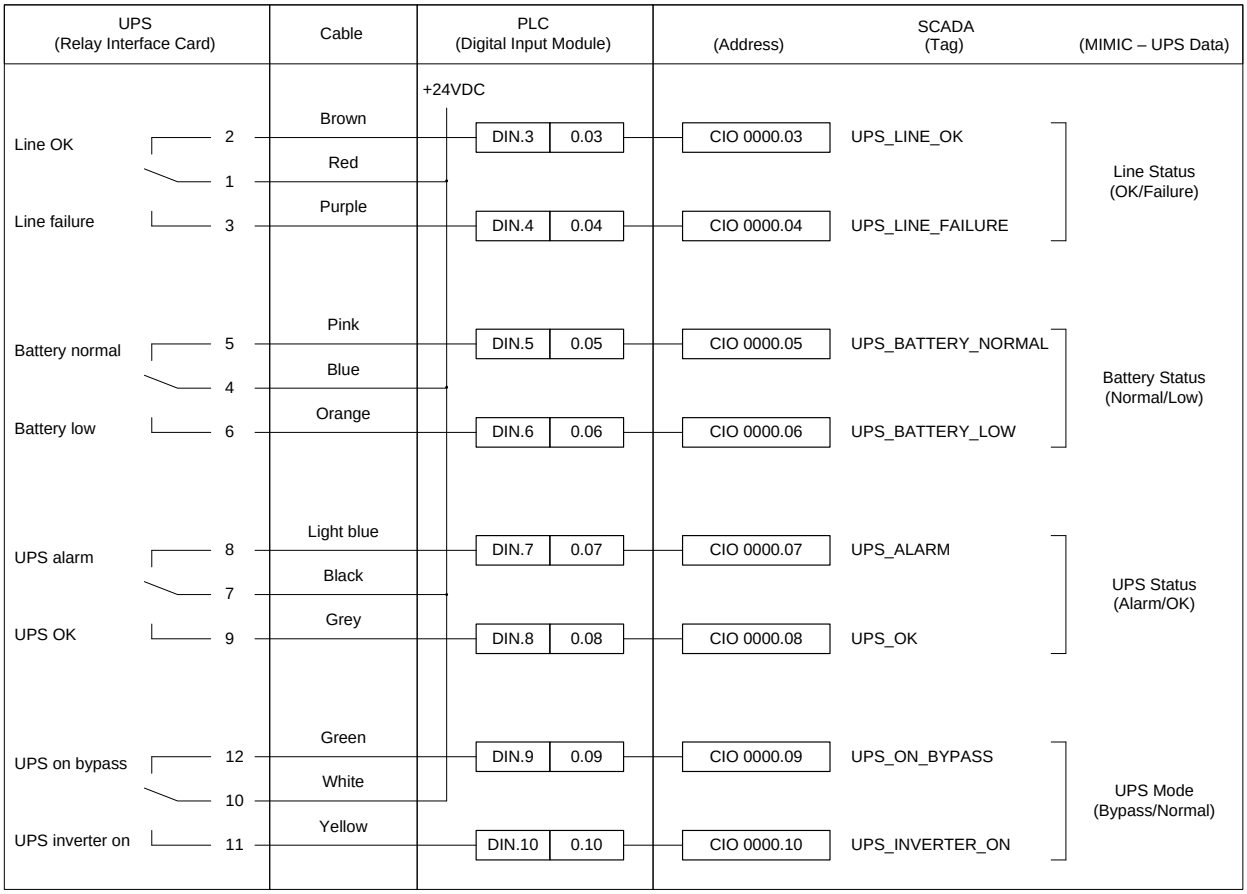
- Set objExcel = Nothing

Transfers the cell data to the string agent that is used to display the value on the SCADA mimic.

- Adroit.SetTag "WEATHERLOCALPRESSURE.value", local_pressure

APPENDIX I

UPS COMMUNICATION INTERFACE DIAGRAM



APPENDIX J

UPS RELAY INTERFACE CARD SPECIFICATION AND FUNCTIONALITY

Relay Interface Card for Powerware 9120 and 9170+

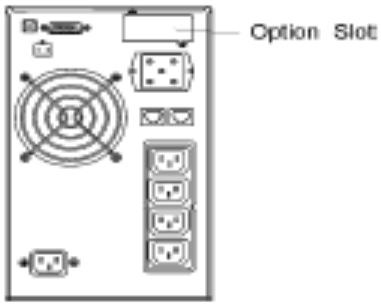
General

This Relay Interface Card (p/n 1014018) provides potential free true relay interface for AS/400 and other relay connected computers and industrial applications. This relay interface uses a 15-pin D-sub connector and this Relay Interface Card is installed in the slot on rear panel of the UPS.

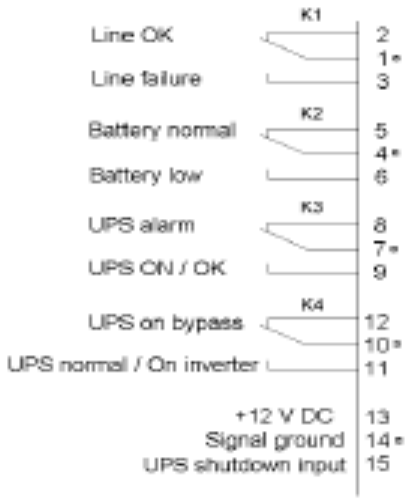
Installation

The UPS does not have to be shut down when installing this Relay Interface Card.

1. Remove the cover on the option slot on the back of UPS to install the Relay Interface Card.
2. Put the Relay Interface Card into the slot and attach it to the rear panel by using the screws removed.



The following information is available from the Relay Interface Card:



* = Galvanically isolated common.

1018946 Rev. A
1/2

Table 1. Relay information available.

Relay # on board	System state	D15 pin # connected	NC / NO *
K1: Line	Line OK	1 - 2	NC
	Line failure	1 - 3	NO
K2: Battery	Battery normal	4 - 5	NC
	Battery low	4 - 6	NO
K3: UPS Alarm	UPS alarm	7 - 8	NC
	UPS ON / OK	7 - 9	NO
K4: Bypass	UPS on bypass	10 - 12	NC
	UPS normal / On Inverter	10 - 11	NO
	UPS shutdown ""	13 - 15	

Table 1

* Normally closed (NC) /open (NO) connection state when the Relay Interface Card is not powered.

** The shutdown pin (15) needs minimum of 5 seconds high level signal (connection to +12VDC) to perform the UPS shutdown.

NOTICE: The relay contacts must not be galvanically connected to any mains connected circuits. Do not use +12VCD (Pin13) of the Relay Interface Card for any power supply purposes for external devices. Reinforced insulation to the mains is required for equipment and cables connected to these connections.

The relay contacts are rated for maximum 1A/30VAC or 200mA/60VDC. All relay outputs are galvanically isolated from the other circuits of the UPS (IEC 950, EN 50091-1).

Shut down function

The Relay Interface Card shut down functionality depends on the jumper (JP1, JP2 & JP3) selected on the board and on the UPS mode (Normal or On Battery). The functions are described in table 2.

Table 2. Shut down options.

		JP1 (factory default)	JP2	JP3
PW 9110	UPS Normal	-	-	-
	On Battery	-	UPS Off immediately, back online when power returns.	-
PW 9120	UPS Normal	-	Immediate output off, back on line when shut down pins are opened	Immediate output off
	On Battery	Shut down as in RS-232 port. Output off after 120 sec, back on line when power returns.	Immediate output off, back on line when shut down pins are opened	Immediate output off
PW 9170+	UPS Normal	-	-	Immediate output off
	On Battery	-	Immediate output off	Immediate output off

Table 2

APPENDIX K

GSM COMMANDER UNIT HOUSING

EASYHOLD

FASTENING SYSTEMS

Slide On Cover Box

Code EHJ5

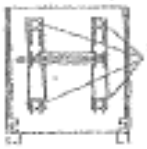
- EASY ACCESS
- WATER RESISTANT
- MULTIPURPOSE APPLICATIONS

Ideal J/Box for:

- MOUNTING ISOLATION SWITCHES
- FIBRE OPTIC JOINTING

Remove lid by lifting HOLDING TABS at the knee bottom and APPLYING PRESSURE upwards on slide-on cover.

Clean mounting holes on underside of base bottom (you can use a drill bit to do this).



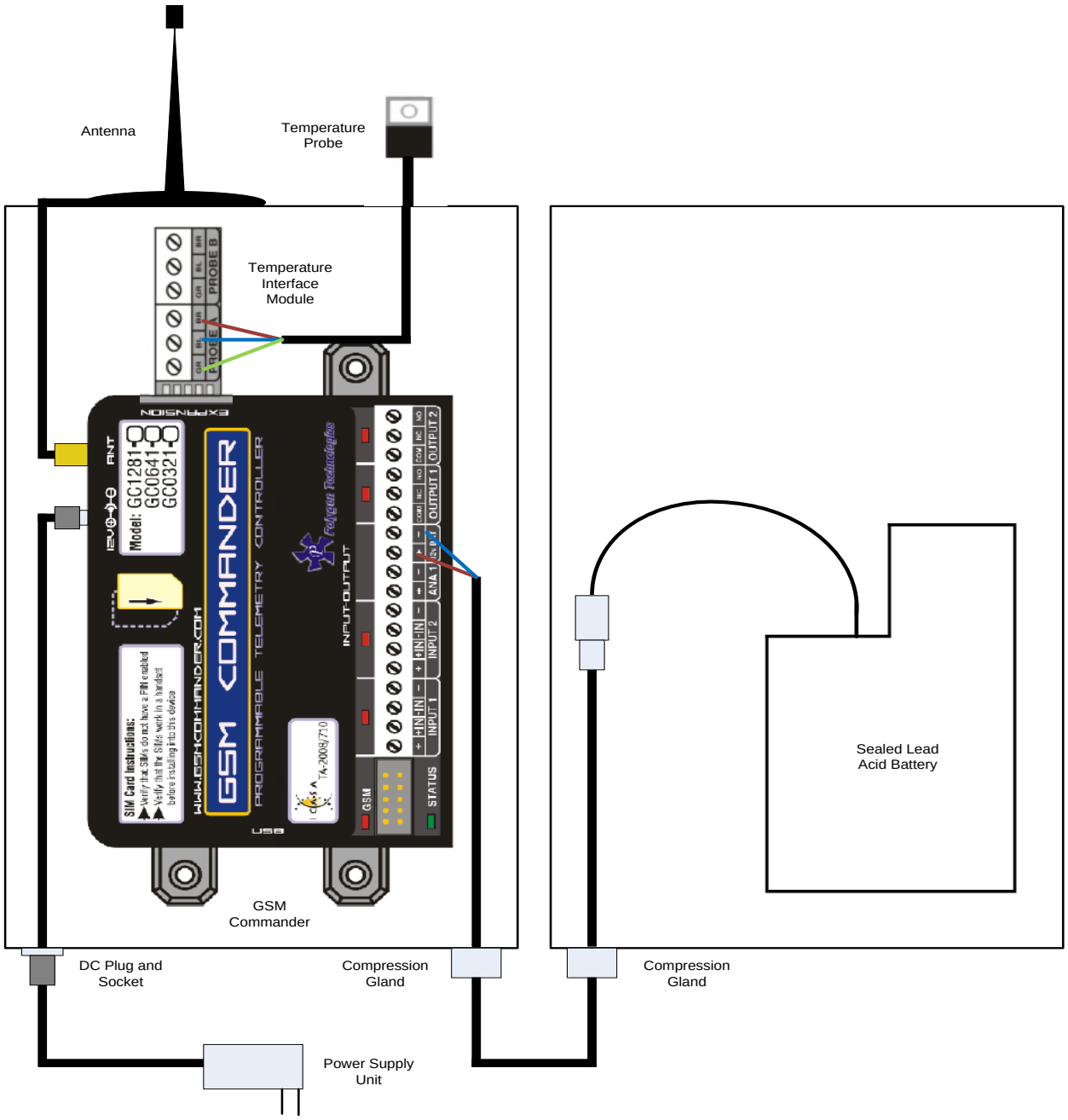
Place base bottom in position and mark mounting holes with pencil. Drill 6 mm hole at marks. Fix base bottom in position by hammering in supplied PLASTIC ANCHORS. If you need to remove box, drill off anchor head, remove base and cut off stumps with STANLEY BLADE.

Secure cover by sliding on till it clips on holding tabs.

* The holes in the MIDDLE are for securing INDUSTRIAL ISOLATOR and can be used for securing objects in box.

APPENDIX L

GSM COMMANDER GENERAL ARRANGEMENT AND SETUP



The components used in the general arrangement of the GSM Commander system is as follows:

- GSM Commander Lite with GPRS (GC0321)
- Antenna
- Power supply unit (PSU1405T – AC100-240V/14VDC, 500mA)
- Temperature interface module (GT002)
- Temperature probe (0-100degC, GT001-1)
- Sealed lead acid battery (0.8Ah, 12V, AS0.8-12)
- SIM card (pay as you go, 0717157019)
- Easyfix isolator enclosure
- PVC compression glands (size 0)
- DC plug and socket (2.1mm)

To successfully operate the system it requires some basic setup as described below:

- Re-charging the system with airtime

It is important the system has sufficient airtime or the SMS functionality will not work. To be able to re-charge the system, purchase an MTN airtime voucher and then dial: *141*voucher number*0717157019#.

- Updating the date and time

When there is mains power failure and the backup battery supply to the unit fails, the unit will switch off. When the mains power is restored, the unit will switch on and send an SMS to the

administrator “Please send “SETTIME” so I can set my time (1)”. When the administrator receives this SMS, the message must be returned to the unit so that the GSM Commander unit can set its date and time correctly. When the unit receives the SMS from the administrator, it will set the date and time and send an SMS back to the administrator “Date&Time set to dd/mm/yy-hh:mm:ss”.

APPENDIX M

CAMERA AND SCADA SETUP PROCEDURE

- Installation

There are two cables connected to the camera namely, a power cable and a network cable.

- Connecting the camera to a power source

The camera has two power inputs of AC24V/DC12V. A 220VAC to 12VDC power supply unit is connected to the camera. The power supply unit is located in the PLC panel.

- Connecting the camera to a local network

The camera has a LAN port. Using a commercial network cable (straight), the camera is connected to a network switch.

- Assigning an IP address to the camera

The camera is connected to the network. A new IP address had to be assigned to the camera when it was installed for the first time.

  - Factory settings

    The factory setting of the camera is as follows:

- IP address: 192.168.0.100
- Subnet mask: 255.0.0.0

o Web browser

To connect to the camera, start Internet Explorer (Web browser), and type the IP address of the camera in the URL box.

o Welcome page

The welcome page of the network camera is displayed in the Web browser.

o Main viewer

By clicking on Enter button on the welcome page, the main viewer is displayed.

o Settings window

The camera functions can be set by the administrator via the settings window. By clicking on the setting button on the main viewer page the following dialog appears.

Enter the user name “admin” and password “admin” and click on OK. The administrator mode menu appears in another window.

o System menu

By clicking on the system tab on the administrator menu, the system menu appears.

This menu will be used to configure the cameras date and time. The system menu consists of five tabs namely; System, Date and Time, Initialize, System Log and Access Log. Select the Date and Time tab. Under Adjust, select the “Synchronize with PC” radio button. Now click on the OK button to validate the settings.

o Network menu

By clicking on the network tab on the administrator menu, the network menu is displayed.

This menu will be used to configure the network settings of the camera to connect the camera to the network. The network menu consists of three tabs namely; Network, PPPoE and Dynamic IP Address Notification. Select the Network tab.

MAC address:

- This displays the MAC address of the camera.

Select the “Use the following IP address” radio button. This is used when a fixed IP address is used.

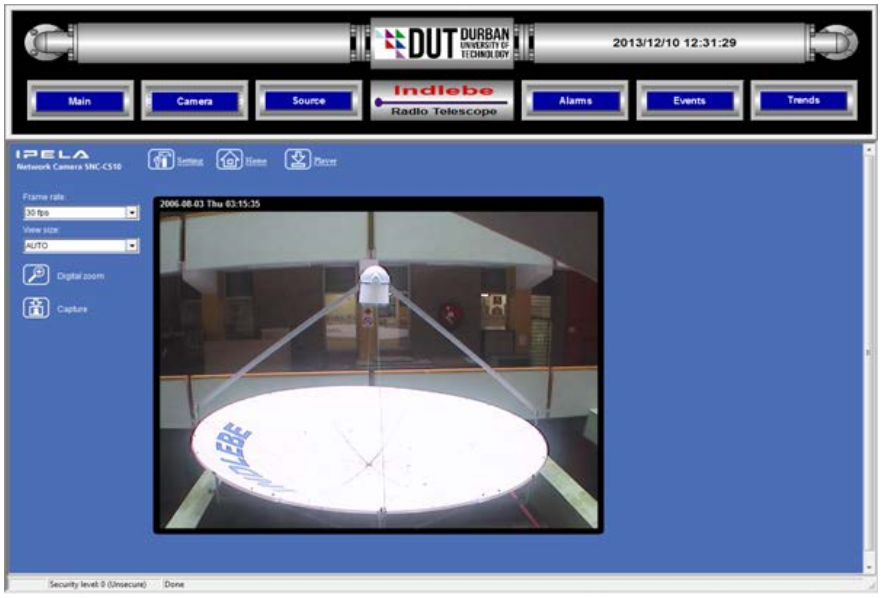
- IP address: change 192.168.0.100 (default) to 169.254.102.3
- Subnet mask: change 255.0.0.0 (default) to 255.255.255.0

Now click on the OK button to validate the settings. Test the new IP via the web browser (refer to section 7.5.2).

- Connecting to the camera from the SCADA system

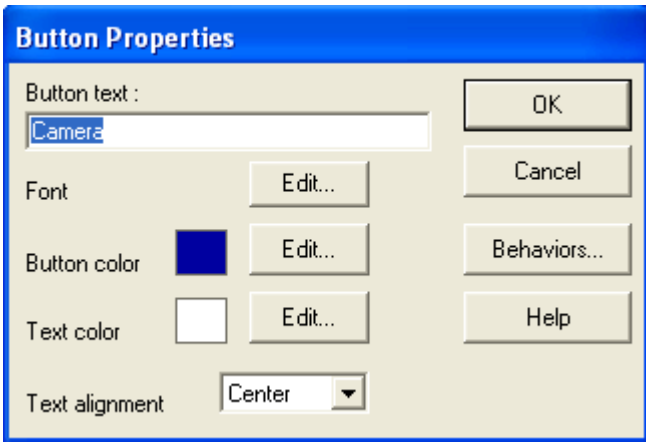
  - Header menu

  Click on file, open and select the header menu file. Click on the camera button on the header menu to display the camera.



  - Camera button properties

  Click on the camera button, then select arrange, beak cell. Double click on the camera button to bring up the button properties.



o Camera button behaviors

Click on the behaviors button in the buttons properties window. The associate behaviors window is displayed. For current behaviors select “execute command”. Select edit to setup this behavior with the following settings:

- Command: open window
- Window type: web
- File name: select the file from the specified directory
- Other settings as per default.

Accept all changes by clicking on the OK button. Select the entire camera button by placing a block around it. Select arrange, make cell. Select file close and save the changes made to the header file.

- Web window

Click on file, open and select the camera web file. Click on view, properties to open the web properties window. Select the web details tab and select the following settings:

- Type in a URL or select an HTML file: http://169.254.102.3/en/AViewer.html
- Refresh speed (sec): 0.00

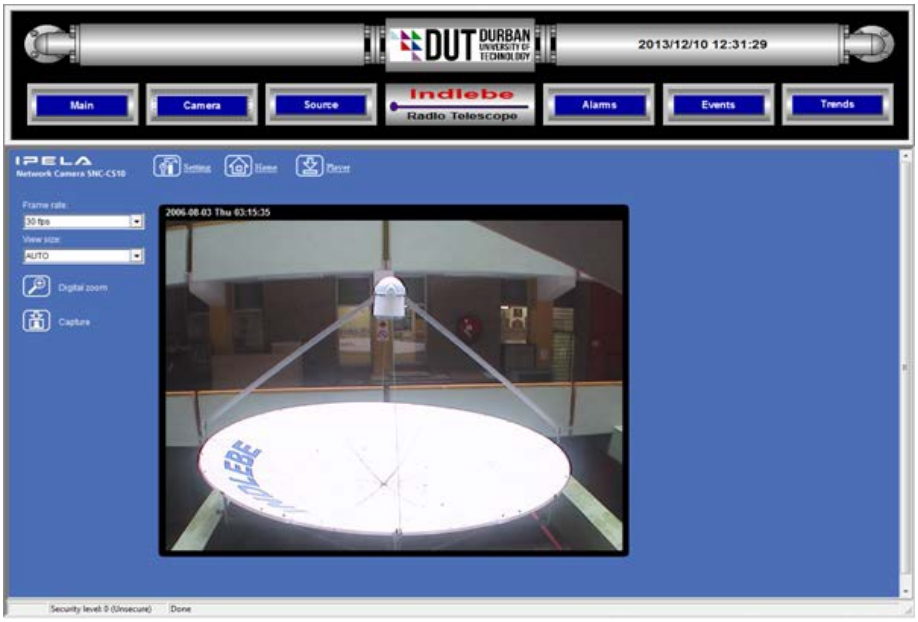
Under the windows tab, select the following options:

- Deselect all options.

Select OK to accept all changes. Select file close and save the changes made to the web file.

- Camera viewing

Click on file, open and select the header file. Click on camera button. This will automatically display the camera view of the antenna.


DUT
DURBAN UNIVERSITY OF TECHNOLOGY
2013/12/10 12:31:29
Main
Camera
Source
indlebe
Radio Telescope
Alarms
Events
Trends
IPELA
Network Camera SNC-CS10
Setting
Home
Preset
Frame rate
30 fps
View size
AUTO
Digital zoom
Capture
2006-08-03 Thu 03:15:35
INDLEBE
Security level: 0 (Unsecure)
Done