ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-TUPDP073

CAN MONITORING SOFTWARE FOR AN ANTENNA POSITIONER EMULATOR

V. van Tonder*, G. Adams, M. Welz
South African Radio Astronomy Observatory, Cape Town, South Africa

Abstract

The original Controller Area Network (CAN) protocol, was developed for control and monitoring within vehicular systems. It has since been expanded and today, the Open CAN bus protocol is a leading protocol used within servo-control systems for telescope positioning systems. Development of a CAN bus monitoring component is currently underway. This component forms part of a greater software package, designed for an Antenna Positioner Emulator (APE), which is under construction.

The APE will mimic movement of a MeerKAT antenna, in both the azimuth and elevation axes, as well as the positioning of the receiver indexer. It will be fitted with the same servo-drives and controller hardware as MeerKAT, however there will be no main dish, sub-reflector, or receiver. The APE monitoring software will receive data from a variety of communication protocols used by different devices within the MeerKAT control system, these include: CAN, Profibus, EnDAT, Resolver and Hiperface data.

The monitoring software will run on a BeagleBone Black (BBB) fitted with an ARM processor. Local and remote logging capabilities are provided along with a user interface to initiate the reception of data. The CAN component makes use of the standard SocketCAN driver which is shipped as part of the Linux kernel. Initial laboratory tests have been conducted using a CAN system bus adapter that transmits previously captured telescope data. The bespoke CAN receiver hardware connects in-line on the CAN bus and produces the data to a BBB, where the monitoring software logs the data.

INTRODUCTION

The MeerKAT radio telescope consists of 64 Gregorian offset antennas, located in the Karoo desert of South Africa. Any individual telescope can move in both the azimuth and elevation axes to direct the telescope towards a specific astronomical source. In the azimuth axis, the telescope can move from -185° to +275°, where 0° points towards the North. In the elevation axis, the telescope can move from 15° to 9°. The azimuth slew rate is 2°/s with 1°/s^2 acceleration whereas the elevation slew rate is 1°/s with 0.5°/s^2 acceleration. A cable wrap exists for rotation in the azimuth axis. Each telescope is fitted with a rotating turret with four slots for different receivers. Currently three receivers are fitted onto the turret making the telescope receptive to UHF, L, and S-band frequencies.

The telescope makes use of a servo-control system that has been developed by CPI VERTEX ANTENNENTECHNIK GmbH (VA) [1]. The control system consist of an Antenna Conditioning Unit (ACU) and Lenze 9400 driver units. The driver units provide current for the motors. The azimuth drive assembly consists of two servo motors that are electrically torqued to avoid backlash. The controller uses three loops to achieve tracking: current, velocity, and position. There are various communication protocols involved in the control system including Controller Area Network (CAN), Profibus, EnDAT, Resolver and Hiperface data.

The Antenna Positioner Emulator (APE) project will consist of the same ACU, Lenze driver units, encoders and motors that a MeerKAT antenna is fitted with. For this project a monitoring system is currently being built and the CAN communication bus is the first protocol to be supported.

CONTROLLER AREA NETWORK

We make use of the CANopen standard. The standard CANopen frame consists of an 11-bit frame ID, a remote transmission request (RTR) bit, followed by zero to eight bytes of data. The 11-bit frame ID consists of a 4-bit code to describe the functionality followed by a Node ID. Each device in the CAN network must have a unique node ID [2].

Table 1 summarises the CAN bus node ID's assigned to the different drivers.

Table 1: CAN Bus Nodes

Drive	Node ID
Azimuth 1	1
Azimuth 2	2
Elevation	5
Receiver indexer	9

In the control system, a function code of 0x180 has been assigned to describe sensors. Each of the 0x180+CAN node ID frame consists of the four sensors listed in Table 2.

Table 2: CAN Sensors

Sensor	Data Length	Data Offset
Status	1	0
Motor temperature	1	1
Torque	2	2
Rotor position	4	4

SOFTWARE ARCHITECTURE

The monitoring software will be implemented on a Beagle Bone Black (BBB) which is an ARM based system. The different software components will run as daemonized processes. Each communication protocol exists in its own

* vereese@sarao.ac.za

ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-TUPDP073

entirety and are run as separate processes. The software also consists of a logger component which can be configured to rotate periodically. Both remote and local control mechanisms exist.

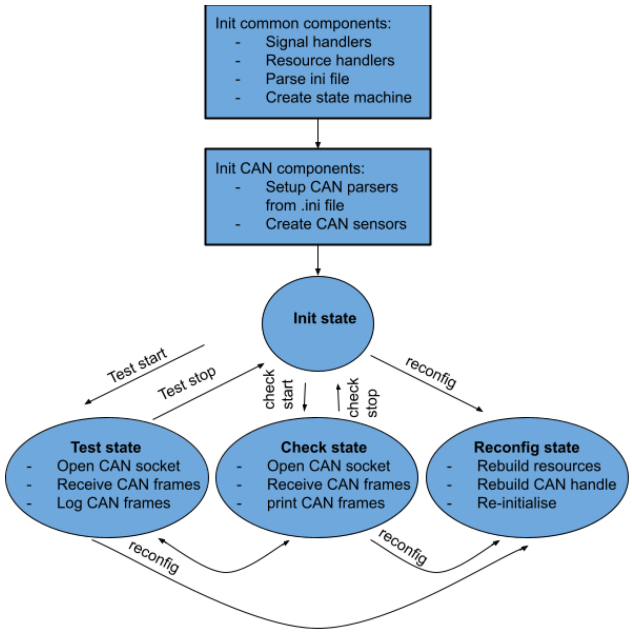

Figure 1: Flow and state diagram.

Figure 1 depicts the flow and state diagram of the CAN monitoring software. Initially the common components are setup. These are components which are available to all the different communication protocols that will be monitored. It includes creating signal handlers, creating resource handlers, parsing the initialisation file, and creating the state machine which starts in an init state. Next, the CAN handler is initialised. This includes creating the CAN frame parsers based on the initialisation file as well as creating the sensors which will be logged.

One can first check that the CAN device is successfully receiving CAN frames by using the ?CHECK START command and state. This will not log any of the sensors but it will display the sensor outputs to screen. The TEST state is activated by initiating the ?TEST START command. This triggers the logger to create the files and starts logging once CAN data is available at the interface. One can move between CHECK and TEST states. The CAN software can be reconfigured whilst the program is up and running using the ?RECONFIG command which will destroy and rebuild all resources, including the CAN data structures. The CAN receiver functionality makes use of the SocketCAN package which ships with standard Linux distributions.

TEST SETUP

Figure 2 depicts the system layout that was used during initial tests. The hardware setup consists of a BBB, two Lenze system bus adapters, custom SARAO CAN receiver hardware, and a laptop. The BBB runs our custom software explained above in order to parse and log the sensors explained in the previous sections.

The transmit code uses text files of real data that was captured on the telescope to simulate a CAN transmitter. The telescope data was captured using a Lenze system bus adapter along with the PCAN-VIEW software developed by Peak systems. Data from both elevation and receiver indexer, which are connected on one CAN bus, are collected whilst moving the antenna in the elevation axis. The receiver indexer is also moved between L- and UHF-band locations. Afterwards, the Lenze system bus adapter was used to collect data in the azimuth axis from both azimuth one and two which is also connected on the same CAN bus. The antenna location is changed and the speed is also increased and decreased.

During the tests, it was noted that the motor temperature was 26 °C. Upon analysing the data, we found a tempera-

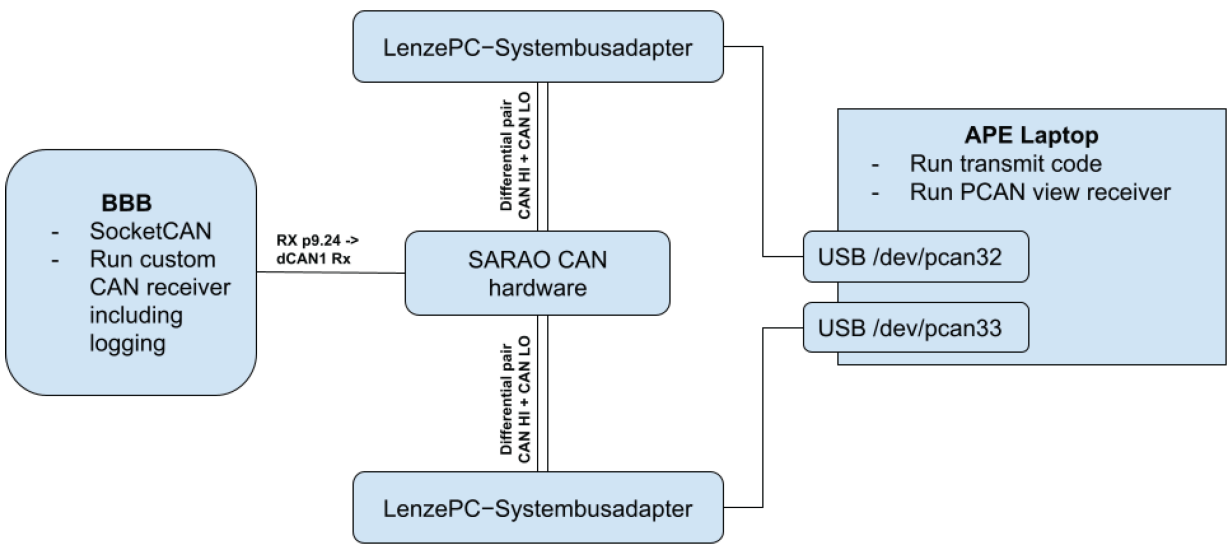

Figure 2: Test setup in laboratory.

ISBN: 978-3-95450-238-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2023-TUPDP073

ture offset of 50 °C between data that was logged using the PCAN-View software and what the system displayed using the Lenze diagnostic tool. Therefore, the linear offset capability was inserted into the monitoring software.

CONCLUSION

The MeerKAT antenna positioner consists of multiple motors and servo drive systems in order to precisely postion the antenna and receiver towards an astronomical source. Various communication protocols exist in order to execute this functionality. This project summarised the implementation and initial tests of custom CAN monitoring software.

The relation between the logged motor temperature and actual temperature has been established to be a 50 °C constant offset. The relations between the actual and logged torque and rotor position remains as future work. Furthermore, the Profibus, EnDAT, Resolver and Hiperface monitoring software needs to be implemented and tested.

REFERENCES

[1] System Description Document: 13.5 m MeerKAT Dual Offset Antenna, Technical report, CPI VERTEX ANTENNENTECHNIK GmbH (VA), Duisburg, Germany, Oct. 2014.

[2] U. Koppe, "TD-03011E Identifier Usage in CANopen Networks", Technical report, MicroControl GmbH & Co. KG, Troisdorf, Germany, 2003. https://www.microcontrol.net/wp-content/uploads/2021/10/td-03011e.pdf