Design and Development of Real-Time Controller Based on ARM Cortex-M Processor for Automated Altitude-Azimuth Telescope Mount

H.A. Rangana Madushanka Perera*, P.G.T.R. Payagala, L.A.D.B.K. Gunawardana

G.D. Illeperuma and V.P.S. Perera

Department of Physics, Open University of Sri Lanka, Nawala, Nugegoda

rangana.rmp@gmail.com

ABSTRACT

Telescopes are widely used in Observational Astronomy. Correctly pointing a telescope towards a desired celestial object requires domain specific skills. Automated telescope mount can be used to improve the accuracy and ease of the hectic process involve with manual position. Real-Time controller which built on Arm Cortex-M4 Processor can be used for an effective and precise positioning of the Telescope. Real-Time nature of the firmware written to the Cortex-M4 microcontroller make the system more reliable and predictable. Inbuilt Digital Signal Processing (DSP) engine and the Floating Point Unit (FPU) in the microcontroller can handle calculations and Digital filter operation with more accuracy. By assigning different priority levels to interrupts and with the help of inter thread communication mechanisms, Soft real-time capability of the system was added. By combining both hardware capabilities and real-time capability of the firmware, desired controlling was achieved.

Keywords: Real-time System, Arm Cortex-M4, Telescope Controller, Altitude-Azimuth Telescope Mount, Observational Astronomy

01. INTRODUCTION

Telescope can be identified as a vital piece of equipment when it comes to observational and armature astronomy. When it comes to telescope tube, it can be fall in to different types like Refractor, Newtonian Reflector, Schmidt-Cassegrain Catadioptric and Maksutov-Cassegrain Catadioptric. The main different among the tubes falls in the way of mirrors and the lenses are

arranged. In order to properly move the telescope tube towards desired location and to place the tube properly, Telescope mounts are being used. Telescope mount consists of two parts, the tripod and the arrangement to hold the tube. [1]

For a person who is relatively new or even for an experience observer, pointing the telescope towards a desired celestial object can be hard and tedious. Due to the relative motion of the celestial objects, it requires an expertise in reading star maps, usage of compass and even knowledge on constellations as well. For a manual Telescope finding a celestial object is more likely a trial and error approach even with the proper tools. Yet another problem with observational astronomy is finding an observational window which is sometimes hard. Some incidents occur once every hundred years, yet observation may be bound by various other factors like weather, Cloud condition of the sky and even light pollution. To get the advantage of the timing window, the telescope should be pointed within minimum possible time. This is very much concern in Astro-photography as it requires long expose shots to get sharper images. [2] [3]

To get rid of the complexity involve with the processes of pointing a Telescope towards a desired location, An Automated Telescope Mount can be used. It eases the observation process as such considerable number of celestial objects will just one click away. But for a country like Sri Lanka, buying such an Automated Telescope Mount seems to be impossible due to the high price. Therefore, this work was aimed towards design an automated telescope mount at a low cost.

02. DESIGN AND THE IMPLEMENTATION OF THE SYSTEM

2.1 REQUIREMENT OF THE SYSTEM

Telescope Controller can simply be identified as a hardware and firmware system with dedicated functionalities. Since the mount will be motor driven, the telescope should be able to control motors in precise manner. Furthermore, it should be able to connect to a PC and ability to compatible with widely used protocol like LX200 gives a world of opportunity to work with well-known PC based Software package like Stellarium. Other than that system should be able to get its current location, current time and the orientation as well.

Other than that it should expect the system to behave in a stable and predictable manner. Expected system block diagram is depicted in below Figure 01, below.

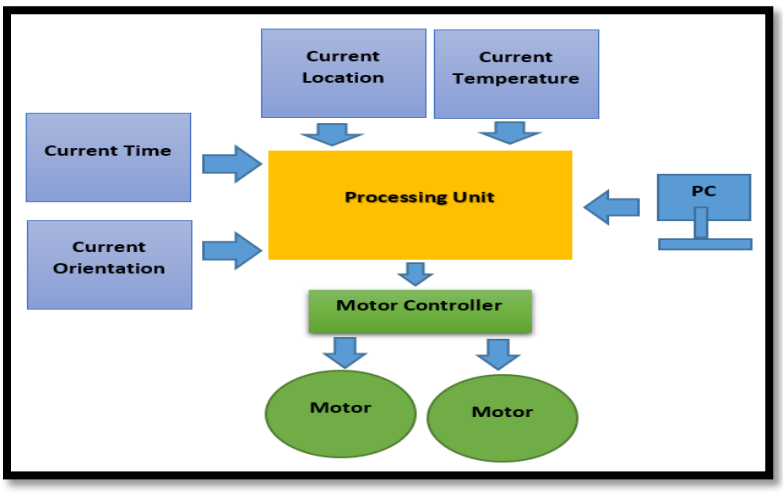

Figure 1- Expected system block diagram

2.2 HARDWARE PLATFORM

Up on reviewing the system requirements, it was decided to go along with a Microcontroller based system. Out of different Microcontroller Architectures available, ARM architecture was chosen due to the fact that ARM based microcontrollers are being developed by many vendors, so there is portability and less learning curve involved. Out of different ARM processors, Cortex-M4 processor was specifically chosen due to many factors like 32-bit addressing, in built Nested Vectored Interrupt Controller, Support for Common Microcontroller Software Interface Standard (CMSIS)-Core, System tick timer and Low power use with respect to performance. Due to the availability of the System tick timer and the Nested Vectored Interrupt Controller, it is possible to write or even port Real-time Operating systems as well [4].For the prototyping phase, STM32F407VGT6 Microcontroller board was used. Actual rotations were carried out using Nema17 stepper motors with 1.80 step angle. TMC series Stepper controllers were used to control the motors precisely.

2.3 FIRMWARE

Initial low level drivers written in C language. “Keil uVision5” was used as the IDE. Keil uVision5 is a powerful IDE with powerful debugging capabilities. After the modules were tested successfully whole code was rewritten to accommodate the Real-time needs of the system. Then the system was tested to ensure the reliability and the predictability as a whole. Figure 2, below

depicts the block diagram of the final system [5].Since one of the main outcomes of the research is to come up with a Real-time system which can cater the functionalities of a Telescope controller. Due to that firmware plays a critical part.

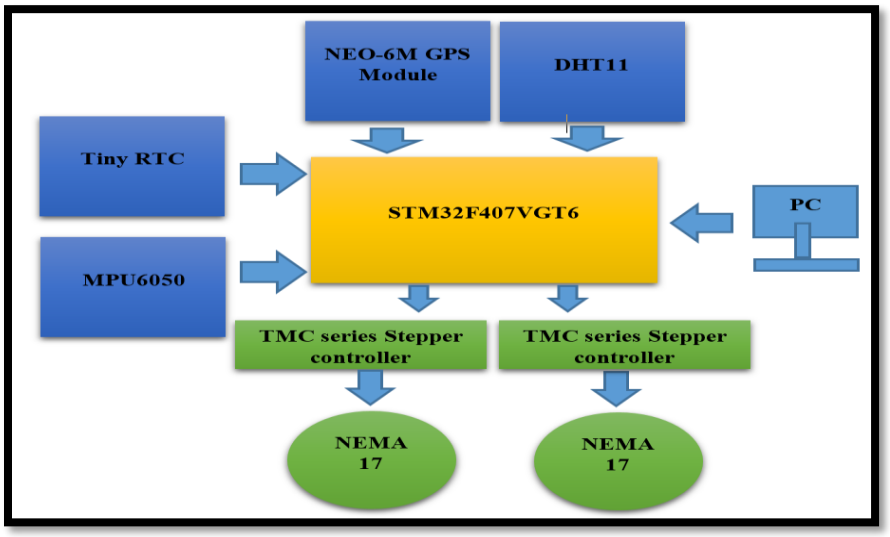

Figure 2 - Block Diagram of the final system

Microcontroller interacts with external events in two ways such as interrupts and poling. The preferred way to interact with external world is interrupts. The main focus when writing the real-time kernel is to organize or prioritize the interrupts to guaranteed the timely and deterministic response. Generally real-time system can fall into either Soft real-time system or Hard-real time system. The main difference is that Hard real-time system's interrupt response time is guaranteed, fail to do so may lead to a catastrophe. Good example is Anti-lock braking system in a car. On the other hand, Soft real-time system can cater an interrupt with some flexible time limit. The build telescope controller falls under the Soft real-time category. [5]

Firmware for a given system can be written in many ways. As depicted in figure 03, one such way is sequential approach. There it will go to next task; one the previous task is being completed. Due to waiting time involved, sensor readings may lose. Next approach is to use threads. Threads simply mean "processor time". As depicted in figure 04, Interrupt service routing (ISR) can be used to manage the threads.

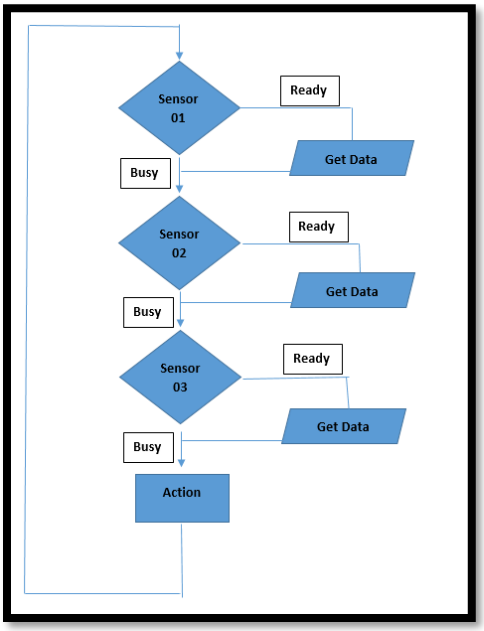

Figure 3 - Sequential Approach

The approach used for this system is, real-time approach. In this approach the processor time is managed by the scheduler. Depending on the priority level, scheduler assign processing time for each task. Figure 5 depicted such a system.

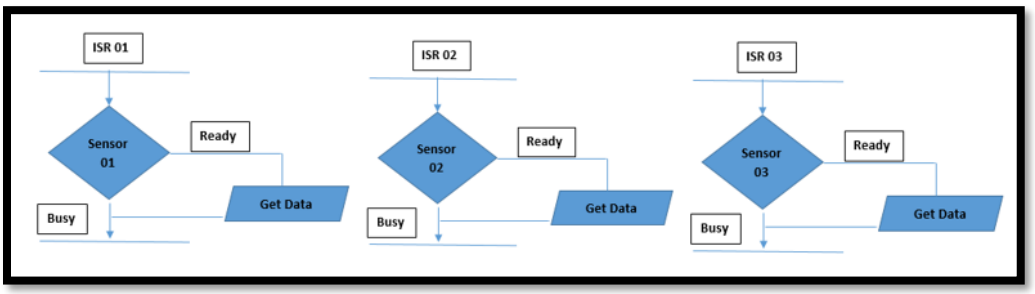

Figure 4 - Using Threads

Real-time capability of the Telescope controller was achieved by using Massage queues and semaphores to intercommunicate independent threads which executed depending on their priority. CMSIS-OS wrapping layer was used where possible to maintain the portability of the code. Inter-task or in between task communications, synchronization between tasks were carried out by Standard Mutexes, Recursive Semaphores, Binary Semaphores and Queues.

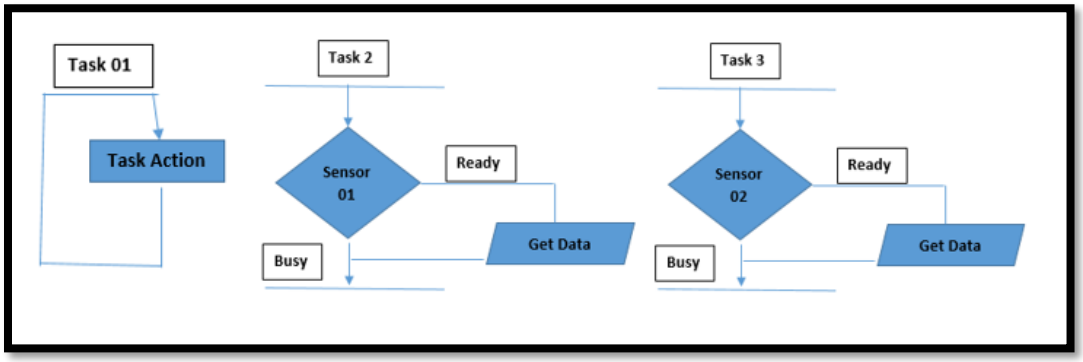

Figure 5 - RTOS approach

01. RESULTS AND DISCUSSION

STM32F407VGT6 Microcontroller based real-time controller which cater the specified requirements was successfully developed and tested. All the sensors which were interfaced was successfully worked with the real-time kernel. The overall system was tested for different input scenarios to ensure the interrupt serving priority and system respond according with the pre assign priorities.

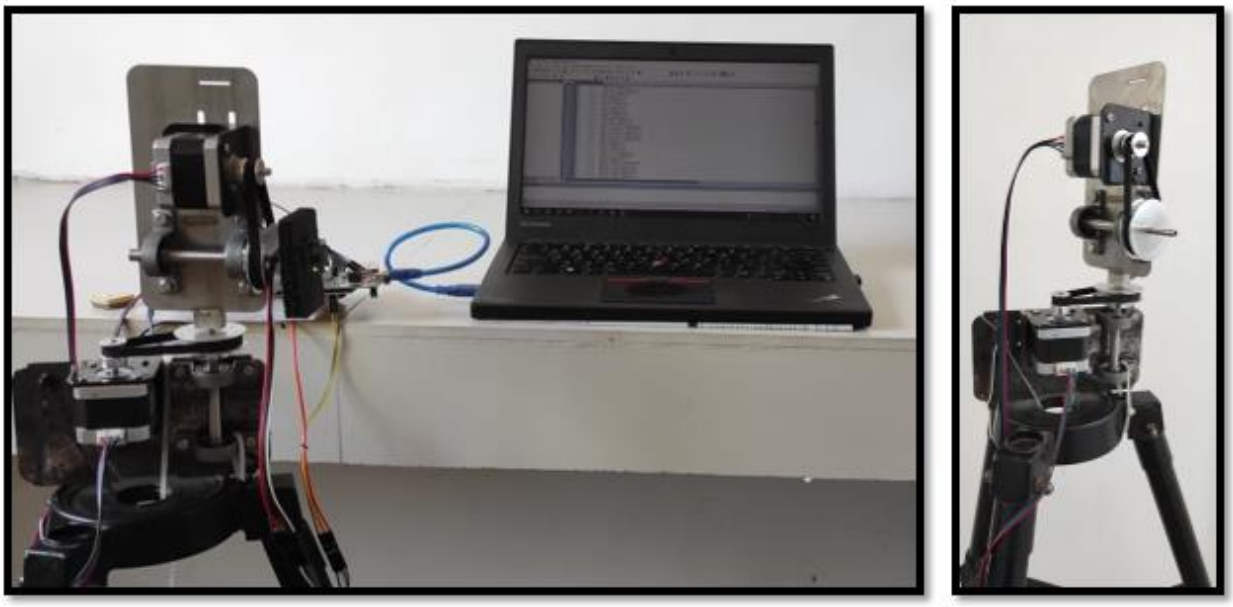

Figure 6 - Final prototype on Left-hand side and the Mechanical test bed on the Right-hand side.

Figure 7 depicted the comparison of the System Tick Time (SysTick), Supervisor Calls (SVC), Interrupt Request (IRQ) and threads.

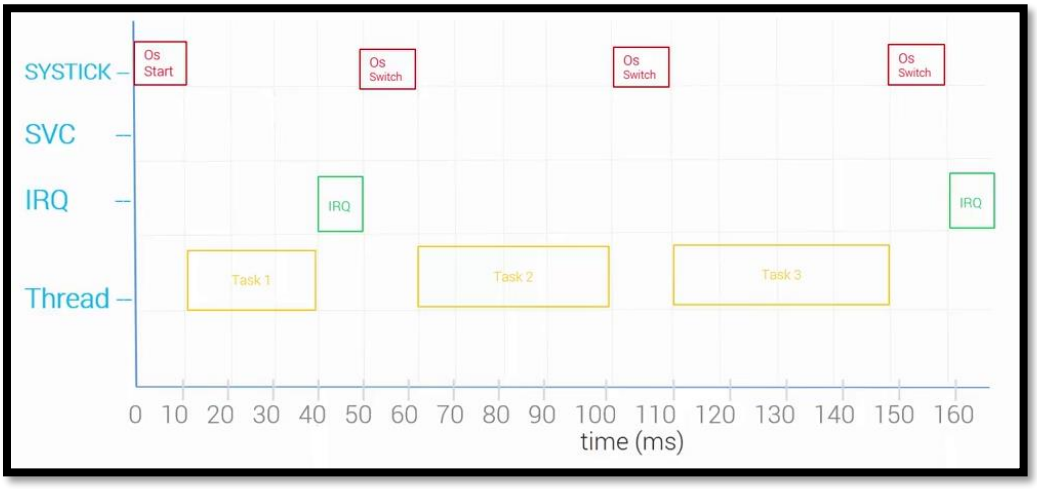

Figure 7- comparison of the system outcomes

02. CONCLUSION

Developed system perform well and achieved the expected outcomes. Since this is going to be a commercial product using Cortex-M4 processor, it can bring many advantages. The main advantage is portability among different vendors. Other than that with the help of inbuilt DSP engine, lot of filtering operations can be achieved without much overhead. Current Real-time system is capable of running a stepper motor in 256 micro stepping resolution mode.

ACKNOWLEDGEMENT

The authors would like to acknowledge the RIC grant under AHEAD project for providing financial support to carry out this research and industry partner for taking steps to commercialize the product.

REFERENCES

[1] X. W. M. W. Jitong Chen, "An Integrated System for Astronomical Telescope Based on Stellarium," in *2011 3rd International Conference on Advanced Computer Control (ICACC 2011)*, Harbin, China., 2011.

[2] M. Swanson, The NexStar User's Guide II, Gewerbestrasse 11, 6330 Cham, Switzerland: Springer International Publishing, 2017.

[3] Xin Li ,Wenlin Zhou,Jun Luo,Junzhang Qian,Wenli Ma, "High Precision Position Control of Telescope Servo Systems," in *IEEE*, Guangzhou, China, 2019.

[4] STMicroelectronics, "High-performance foundation line, Arm Cortex-M4 core with DSP and FPU, 1 Mbyte of Flash memory, 168 MHz CPU, ART Accelerator, Ethernet, FSMC," 13 12 2021. [Online]. Available: https://www.st.com/en/microcontrollers-microprocessors/stm32f407vg.html.

[5] J. W. Valvano, REAL-TIME INTERFACING TO ARM® CORTEXTM-M MICROCONTROLLERS, texas.USA: Jonathan W. Valvano, 2014 .

[6] A. Limited, "ARM Developer," 4 12 2021. [Online]. Available: https://developer.arm.com/.

[7] R. N. B. T. S. Kumar, "Observer Based Control of a Brushed DC Motor at Very Low Speeds," in *11th Asian Control Conference (ASCC)*, Gold Coast Convention Centre, Australia, 2017.

[8] S. Y. T. S. Kumar, "An Overview of IoT Enabled Generic Embedded Controller Developed," in *2019 9th International Conference on Emerging Trends in Engineering*, Nagpur, India., 2019.

[9] F. L. Taoren Li, "Design of Remote Monitoring System Based on," in *2019 IEEE International Conference on Power, Intelligent Computing and Systems (ICPICS)*, Shenyang , China, 2019.

[10] J. Yiu, Definitive Guide to the ARM Cortex-M3 and Cortex-M4 Processors, Kidlington, Oxford OX5 1GB, UK: Elsevier Inc, 2014.