71st International Astronautical Congress (IAC) - The CyberSpace Edition, 12-14 October, 2020.

IAC–20–C3.4.5

DESIGN AND IMPLEMENTATION OF THE SOFTWARE ARCHITECTURE AND CONTROL FOR THE ELECTRICAL POWER SUBSYSTEM OF A 3U HYPERSPECTRAL IMAGING CUBESAT

Parth Kharade[a*], Devaansh Gupta, Dhairya Arora, Neel Tailor, Lavitra Garg

[a] *Birla Institute of Technology and Science (BITS)-Pilani, India, f20170352@pilani.bits-pilani.ac.in*
[*] **Corresponding Author**

Abstract

The paper discusses the conceptualization, design and implementation specifics of the software architecture and control for the Electrical Power Subsystem (EPS) of a 3U CubeSat equipped with a hyperspectral imaging camera as its primary payload and an FPGA as the secondary payload.The On Board Computer(OBC) of the satellite already runs multiple attitude control and maneuvering operations in addition to the command and overall data handling.Given that the satellite needs to execute some more tasks simultaneously the EPS has its own software control for administering tasks related to power management, storage and other interfaces. For this purpose, MSP430F5529LP microcontroller by Texas Instruments is chosen. Being the first subsystem to turn on, EPS is responsible for booting the OBC and deploying the onboard antennas.The microcontroller executes the OBC bootup sequence by powering it up and loading the OS either from the onboard PROM or EEPROM. The process runs continuously till the microcontroller receives a timer reset signal from the OBC. After the OBC has booted the EPS continues to function as an external watchdog. The code for implementing the external watchdog has been successfully tested on the microcontroller.Furthermore, the burn-wire mechanism for antenna deployment has also been evaluated with the antenna deployment module. Considering the power constraints, the EPS implements the Perturb and Observe(PO) Algorithm for Maximum Power Point Tracking (MPPT) of the solar panels. The algorithm has been successfully implemented on the microcontroller. To implement the PO algorithm the microcontroller has to acquire current and voltage data from the solar panels. Furthermore the microcontroller also collects housekeeping data from the rest of the satellite. This data includes current, voltage, temperature parameters and status flags of the battery, solar panel and different satellite subsystems. This data is then transmitted as a simple beacon and provided to the OBC on demand. I2C protocol has been implemented for data acquisition from various sensors to the microcontroller.The EPS also monitors the overcurrent protection circuitry of the satellite and reports to the OBC when any overcurrent fault occurs The paper thus provides a complete and detailed view into the functioning and implementation of an autonomous Electrical Power System in terms of both hardware and software. It also attempts to explain the interaction of EPS with other components, including sensors, power management ICs as well as other on-board controllers.

Abbreviations

- Electrical Power Subsystem,(EPS)
- On-Board Computer,(OBC)
- Over-Current Protection Circuit,(OCPC)
- Microcontroller Unit,(MCU)
- Poly-Pico Satellite Deployer,(P-POD)
- Not Acknowledge,(NACK)
- Acknowledge,(ACK)
- Telemetry, Tracking and Command,(TTC)
- Maximum Power Point Tracking,(MPPT)

1. Introduction

The Electrical Power Subsystem(EPS) is one of the six subsystems of the nanosatellite and one of the two on-board controllers. The satellite employs an asymmetric distributed architecture wherein it has two microcontrollers on-board: the On-Board Computer(OBC) and the EPS microcontroller.
The division of tasks between these two microcontrollers is made on the basis of criticality. The critical

tasks such as power generation storage and distribution are implemented on the low power EPS microcontroller. This enables the satellite to recover from low power conditions. Furthermore, this provides a half redundant system which can recover from any faults that might occur on the OBC.
The paper discusses the tasks of the EPS microcontroller and their on-board implementation. Section 2 lays out the tasks which the EPS microcontroller executes on-board and it's modes of operation. Section 3 discusses each of these tasks in detail. Section 4 presents an overview of how these tasks are scheduled and also contains some latency analysis. Section 5 concludes the paper. Appendix A (Tables) has a table listing the worst-case execution times of all the interrupts in the EPS code. Appendix B (Flowcharts) contains flowcharts for the tasks which have been discussed in Section 3.

2. Tasks and Modes of Operation

This section briefly discusses the tasks which the EPS executes and classifies them into various modes in which the system operates.

2.1 Tasks

1. **Peripheral Initialization**
   Initialisation of all the peripherals such as Timers, I2C and SPI modules, GPIO modules etc.

2. **Antenna Deployment**
   The EPS, being the first subsystem to turn on is tasked with deploying the antennas. It employs a burn wire mechanism to deploy the on-board dipole and monopole antenna.

3. **Simple Beacon**
   The simple beacon acts as a basic link between the ground station and the satellite and is important for indicating the presence of the satellite in orbit.

4. **Maximum Power Point Tracking**
   The hyperspectral imaging payload and the FPGA for on-board image compression pose harsh power requirements. The Electrical Power Subsystem implements an MPPT algorithm to maximise the amount of power extracted from the solar cells.

5. **External Watchdog**
   The EPS acts as an external watchdog for the OBC. The Electrical Power Subsystem power cycles the OBC if it is unresponsive.

6. **Housekeeping**
   The EPS collects power, voltage and temperature housekeeping data from various on-board sensors.

7. **Overcurrent Protection**
   The EPS implements an overcurrent protection circuitry for critical on-board components with the retry logic implemented on the EPS microcontroller.

8. **Communication with the OBC** The EPS communicates with the OBC for exchange of command and data.

9. **Thermal Control**
   The EPS controls the heaters connected to the payload and the battery.

2.2 Modes of Operation

Over the mission duration, the satellite is exposed to various stimuli and must take appropriate autonomous action. This results in the satellite switching between various states with each state corresponding to a defined group of tasks it must execute. We call these states the modes of operation of the satellite. The satellite has 12 modes of operation[1] which have been listed below.

1. **Init mode** : The EPS boots up and is functional.

2. **EPS-OBC boot up sequence** : The EPS boots up the OBC.

3. **Basic mode** : The EPS and the OBC are performing their basic tasks.

4. **Detumbling mode** : The mode in which the main task is to detumble the satellite.

5. **Idle mode** : The TTC boots-up and is operational. All subsystems of the satellite are powered on.

6. **Image compression** : The FPGA performs image compression on the collected images. Increased power comsumption.

7. **Sun-tracking** : Orienting the satellite such that the maximal area is exposed to the sun for increased power generation.

8. **Sun-tracking with image compression** : Increased power generation in mode 7 can facilitate mode 6.

9. **Pointing** : Orienting the satellite to point in a particular direction for ground station communication or image capture.

10. **Payload execution** : Hyperspectral imager operational for image capture.

11. **Downlink** : Downlink of collected data to the groundstation.

12. **Momentum Dumping** : Dump the momentum accumulated by the reaction wheels of the satellite.

The EPS, however will have far fewer modes of operation as the tasks it must perform will be the same for many of the satellite's modes of operations.
The EPS modes of operation have been outlined below.

1. **Init Mode**:
   This mode is entered in the event of a power cycle of the EPS MCU. In this case, the EPS first reinitializes its peripherals and eventually returns to the mode it was previously operating in.
   - Peripheral Initialisation

2. **Deployment Mode**:
   The satellite enters this mode as soon as it is deployed. This mode is entered only once during the life of the satellite.
   - Antenna Deployment
   - Maximum Power Point Tracking.
   - Thermal Control
   - Overcurrent Protection

3. **Idle Mode**:
   This is the mode the satellite enters after init mode or when there isn't enough power to keep the OBC on. The EPS waits in this mode till there is enough energy to reboot the OBC. The EPS starts sending the simple beacon in this mode.
   - Maximum Power Point Tracking
   - Thermal Control
   - Overcurrent Protection
   - Simple Beacon
   - OBC-Bootup (*On availabilty of sufficient energy*)

4. **Normal Mode** :
   This is the normal mode of operation of the EPS when the OBC is up and running.
   - Maximum Power Point Tracking.
   - Thermal Control
   - Overcurrent Protection
   - Simple Beacon
   - External Watchdog
   - Housekeeping
   - OBC Communication

3. Tasks

3.1 Antenna Deployment

The satellite has two deployable antennas: a monopole and a dipole. These need to be deployed for communicating with the ground station. The EPS is the first system to power on after deployment from the P-POD and carries out the antenna deployment procedure. A burn-wire mechanism will be used to deploy the on-board antennas. The antennas are coiled up in a cavity inside the module and exert outward pressure on the gate. The gates are held shut by nylon wires which are wound around a resistor. To deploy the antennas the nylon wires are burnt by passing current through the resistors. On deployment the EPS will start a fifteen-minute countdown timer to the antenna deployment procedure. This is in accordance with CalPoly standards which state that antennas may be deployed 15 minutes after ejection from the P-POD. Feedback for the antenna deployment is taken from switches which will open when the gates have been released. The antenna deployment status is then passed to OBC when it boots-up.

3.2 Watchdog

The EPS acts as an external watchdog for the OBC. This is accomplished through a simple GPIO interface and a timer on the EPS MCU the OBC must send a GPIO interrupt to the EPS MCU to reset the timer before it runs out. In case the timer runs out, the EPS powers down the OBC and after some time boots from a different image/memory.

3.3 Simple Beacon

The simple beacon is the most basic form of communication that the satellite has with the ground station/s. It contains the name and the callsign of the

satellite. The simple beacon will be transmitted in the UHF band at a frequency of 435MHz. The beacon is morse coded and uses a MAX1472 IC for OOK Modulation. The morse code will be stored as an array of 0s and 1s and the microcontroller will traverse the array. The value of the element will be fed as data to the OOK modulator.

3.4 Housekeeping

The housekeeping data consists of thermal, voltage, and power data of various components of the satellite. Currently, we are using the INA219 high side current sensor which also reports power and voltage, and the LM75A temperature sensor with thermal watchdog capabilities. The sensors are interfaced with the microcontroller using the I2C protocol.
The sensors have 16 bit wide registers and report 2 bytes of data. This data is sequentially stored in a single dimensional array. Whenever a sensor responds with a NACK the corresponding bytes are filled with a value that is outside the sensor's reportable range. This can be recognized on the ground station and such data can be discarded.

3.4.1 Maintaining Data Coherency

There are two tasks that access the housekeeping data. The OBC communication task and the housekeeping task. The housekeeping data is stored in a single dimensional array. When the data is being transferred to the OBC the OBC Communication task just sweeps the array byte-wise.
It might happen that the OBC requests data while the housekeeping task is updating it. Such an instance of the housekeeping block will contain some old data and some newly updated data. Mixing of data collected at two separate time intervals can lead to incorrect inferences when the data is analysed.
To solve this a header byte has been inserted at the beginning of each block. Whenever the housekeeping task starts updating a block the byte is set to *invalid* and sets it back to *valid* when it has finished collecting all the data. Hence data accessed by both the tasks simultaneously will have an *invalid* header byte and can be discarded.

3.5 MPPT

The satellite uses the Perturb and Observe algorithm to implement maximum power point tracking algorithm. This algorithm has been chosen because of its simplicity to implement and because it has been previously used by other cube satellites. The MPPT task has the least priority af all tasks. The reasons for this are detailed in Section 4.

3.6 SPI

The communication interface between EPS and OBC is full duplex where EPS will send housekeeping data to OBC and in return will receive commands for switching individual subsystems ON/OFF. SPI protocol was chosen as the I2C modules of the OBC were interfaced to various sensors and already used up. EPS has been configured as a slave device in the 4 Pin SPI mode while the OBC is configured as a SPI master device.

3.6.1 Distinguishing between Command and Garbage Data

As discussed earlier the OBC can initiate an SPI transaction with EPS for two reasons.

1. To request housekeeping data.
2. To send a switching command.

As SPI is a full-duplex protocol data is both simultaneously sent and received but the bytes received by the EPS while sending housekeeping data are not OBC commands. If they are interpreted as such this will lead to erroneous switching of components. To distinguish between actual commands and garbage data a GPIO pin and a flag variable is used. The status of the GPIO pin indicates if the data being sent is a command or a garbage value. lets call this pin *transaction mode pin* and the flag variable *command* This logic is handled through the transmit and receive interrupts of the SPI module. For a byte of data exchanged the transmit interrupt always occurs before the receive interrupt.

3.6.2 Transmit Interrupt

When a SPI transmit interrupt occurs the MCU first checks the status of the transaction mode pin.

- If the pin is low it indicates that the OBC has initiated a housekeeping transaction. In this case the command flag is kept at *false* and the next housekeeping data byte is loaded into the Transmit Buffer and the housekeeping pointer is incremented.
- If the pin is high, it indicates that the OBC is sending command data in the current transaction. The command flag is set to true, the housekeeping pointer is reset to 0.

3.6.3 Receive Interrupt

In the receive interrupt the MCU checks the status of the command flag.

- If the command flag is true, the data is read as a switching command and the appropriate component switching is carried out.

- If the command flag is false, the data in the Receive Buffer is not read and the execution exits the ISR.

3.7 OCPC

In space, components are prone to Single Event Latch-Ups and a power cycle is necessary to correct such situations. Without an overcurrent protection circuit, the component may incur irreversible damage which might render it unusable. Conductive paths formed by SELs can be remedied by power cycling the affected component. The EPS uses Texas Instruments' TPS2553-1 power switch to implement this functionality.
In the event of an overcurrent, the IC cuts-off the power supply to the component and informs the microcontroller via a GPIO interrupt.
The microcontroller then turns on the switch after a few milliseconds. If an IC gets permanently damaged and frequently trips the power switch in a small amount of time, it is turned off permanently to prevent a power cycle loop. For the EPS microcontroller itself, overcurrent circuitry is handled in the hardware with multiple auto retry functionality. Delay is introduced with the help of RC circuits.

3.8 Thermal Control

The EPS implements active thermal control for the Payload and the battery. It uses Kapton®heaters for heating and LM75A temperature sensors for feedback. The LM75A has thermal watchdog capabilities and can generates over-temperature and under-temperature interrupts.

- When the MCU detects an under-temperature interrupt it turns on the Kapton®heaters for the affected components. The temperature is periodically monitored through the housekeeping task and the heaters are turned off when a sufficient temperature has been achieved.

- When the MCU detects an over-temperature interrupt the component is disconnected from the power supply to reduce heating. This is the best plan of action in the absence of an active cooling system

4. Scheduling and Latency Calculation

4.1 Scheduling

All the tasks that the EPS executes are simple in nature and can be easily scheduled without an RTOS. A timer is used for generating timing signals for periodic tasks such as Housekeeping, MPPT and simple beacon transmission, while other event based tasks such as OCPC,External Watchdog, Thermal Control, OBC Communication are triggered by the Interrupt Service Routines corresponding to the relevant events.
Then based on the complexity of the task, the task is either handled inside the ISR itself or a flag is set which triggers the execution of the tasks in the main() function.
It must be noted that the MSP430 has fixed priorities for interrupts which cannot be changed by the user. This means that allocation of peripheral modules to the tasks must be in accordance with the priority of the task. Furthermore, the MSP430 does not have inherent support for interrupt nesting. Although this can be manually enabled, it is not advised due to the limited SRAM capacity of 8KB.[2]
The MPPT task was found to have a longer execution time as compared to the other tasks. This was since the algorithm must wait for the sensor data before proceeding. Furthermore, the algorithm itself was too long to fit inside an ISR. Hence, the MPPT algorithm is executed as a function within the main while loop. This function is periodically triggered by a timer. This MPPT task becomes the lowest priority task of the system which can be pre-empted by all other tasks of the system. This isn't an issue because the pre-emption is expected to happen less frequently. The MPPT task runs with a time period of 10ms and is expected to be pre-empted once or twice every 5s by the housekeeping in the worst-case scenario. Even when the MPPT task is pre-empted it is unlikely that it's normal operation will be affected.
All the other tasks have been split into smaller subtasks which are triggered by specific interrupts and the sub-tasks have been implemented inside the ISRs of the interrupts which trigger them.
Execution times of all the ISRs are given in *Appendix A (Table)* The execution times are found to be within 100us which is within the tolerable limits. The execution time can further reuduced by increasing the clock speed of the MCU.

5. Conclusion

The paper explains the software architecture for an Electrical Power Subsystem in a distributed architecture. It also divides the operation of the system into modes and tasks. The paper discusses the design and implementation of these tasks in detail taking into consideration the resources available on the microcontroller. All of these tasks have been implemented and verified on the actual microcontroller board. The paper also provides a scheduling scheme for the tasks by breaking them into interrupt service routines. A timing and latency analysis was conducted on the interrupt service routines to make sure that the tasks don't miss their deadlines and are executed with tolerable latencies.

5.1 Scope for Improvement and Future Work

There is scope for reducing the execution time of the current code by increasing the clock speed of the MCU. However increasing the clock speed will lead to increased power consumption. This trade-off will have to be analysed. Sleep modes can be implemented during the time the microcontroller is inactive. While implementing sleep modes will give reduced power consumption the added latency due to the wake-up time will have to be considered.

6. References

[1] R. Jain et al., Modes of Operations for a *3U CubeSat with Hyperspectral Imaging Payload*, 2018 International Astronautical Congress
[2] Texas Instruments, *MSP430 Family User Guide*, Revised-July 2013

Appendix A (Table)

	Task	Associated ISRs	Operations to be performed	Worst Case Execution Time(CPU Cycles)
1	Antenna Deployment	15 minute timer expiry	Turn on MOSFETS for antenna deployment	34
		Feedback from gate	Set gate deployed status -> TRUE	25
2	Simple Beacon	Timer interrupt to send next bit	Set GPIO pin HIGH or LOW depending on the bit	45
4	Housekeeping	Timer Interrupt to Begin Housekeeping Collection	Generate Start condition on the I2C bus	56
		Byte Received Interrupt	Read byte into memory.	96
			Condtionally increase necessary pointers and generate I2C start conditions	
		Byte Transmit Interrupt	Load next byte to be transmitted into the transmit buffer	96
			Conditionally generate I2C start in receive mode	
5	OBC - Watchdog	Watchdog Timer Expired	Shutdown OBC. Start Countdown Timer for reboot.	
		Timer Reset Interrupt from OBC	Reset Timer	30
		Reboot Timer Expiry	Reboot OBC from new memory.	
6	OBC Communication	Byte Transmit Interrupt	Load next byte to be transmitted into the transmit buffer	26
		Byte Received Interrupt	Read byte and conditionally toggle output on GPIO pins to switch Components.	26
7	Overcurrent Protection	Overcurrent Fault Interrupt	Set Enable pin of TPS2553-1 to low.	38
			Start timer for component reboot.	
			Loop timer expiry - Check number of faults conditionally disable component.	40
8	Thermal Control	Overtemperature Interrupt	Turn off the heaters via GPIO signals	34
		Undertemperature Interrupt	Turn on the heaters via GPIO signals	34

Table 1 : Gives details of all interrupts and their worst case execution times in terms of CPU Clock Cycles

Appendix B (Flowcharts)

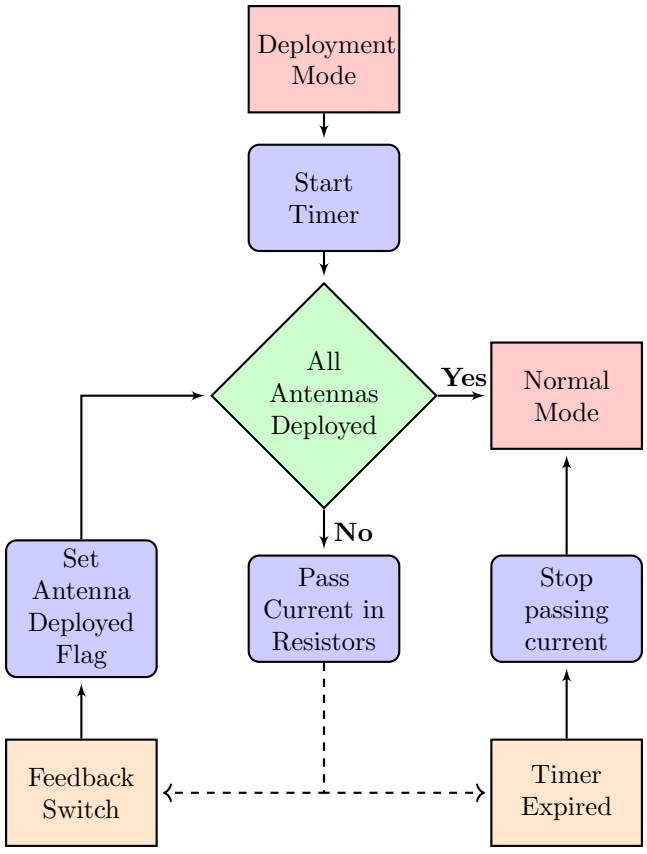

Fig 1 : Antenna Deployment Task

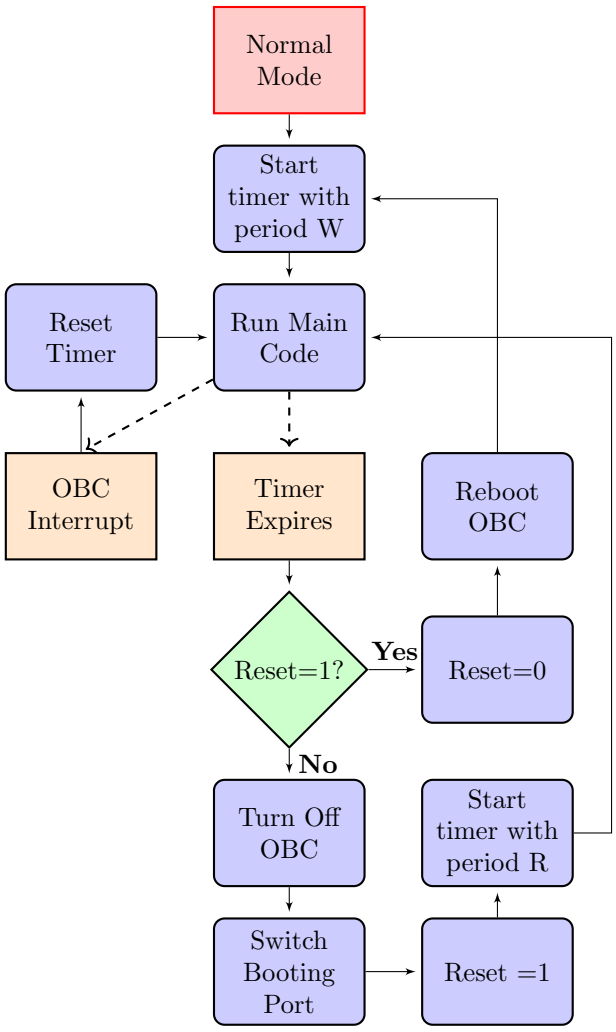

Fig 2 : On-Board Computer Watchdog Task

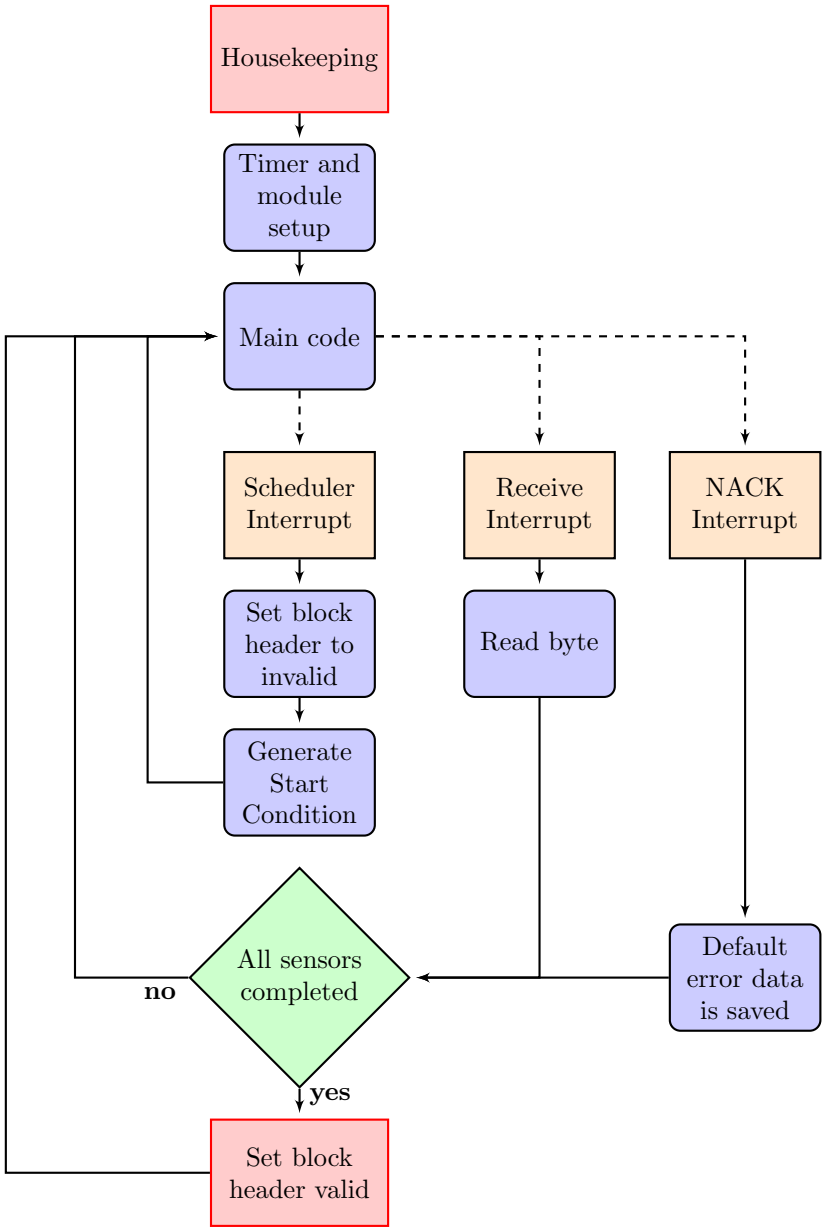

Fig 3 : Housekeeping Data Task

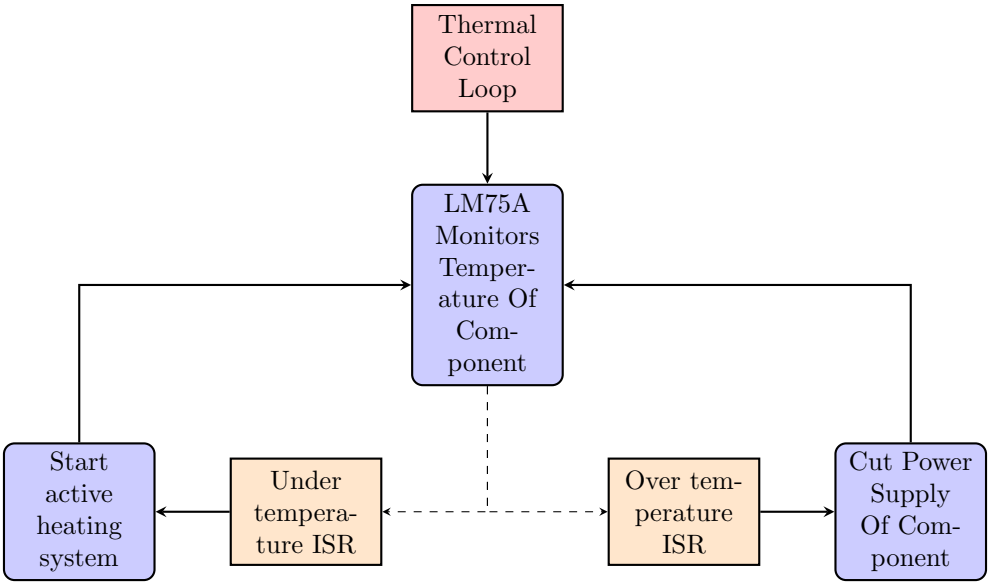

Fig 4 : Thermal Control Task

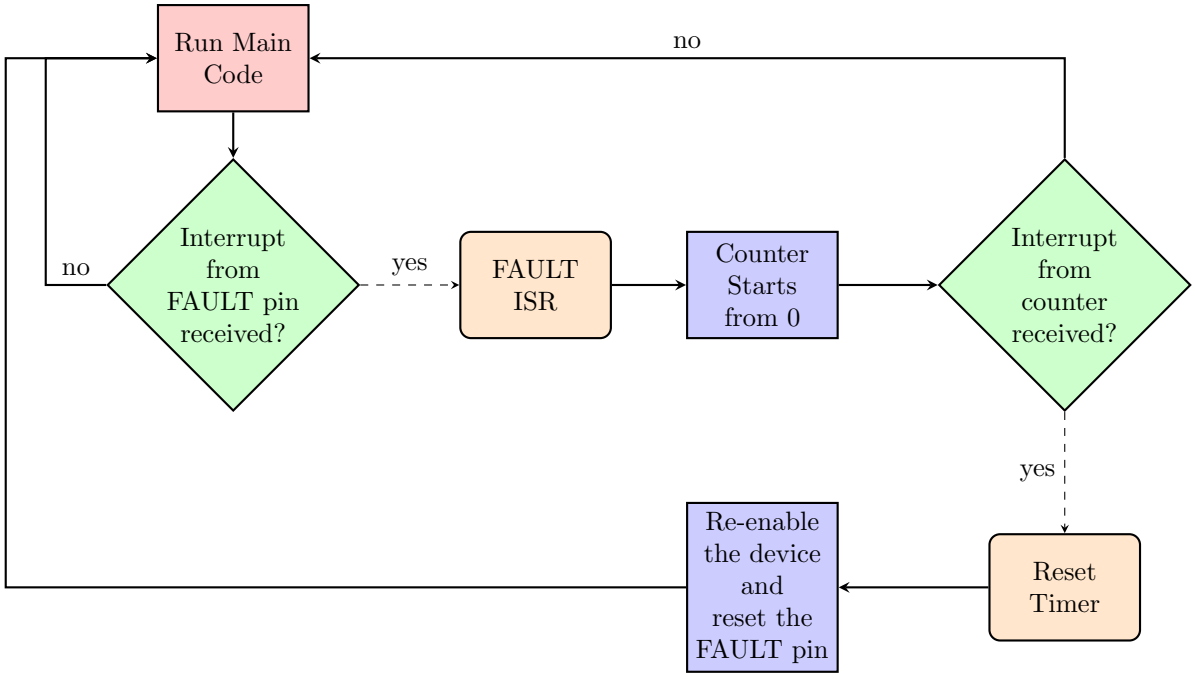

Fig 5 : Over-current Protection Task

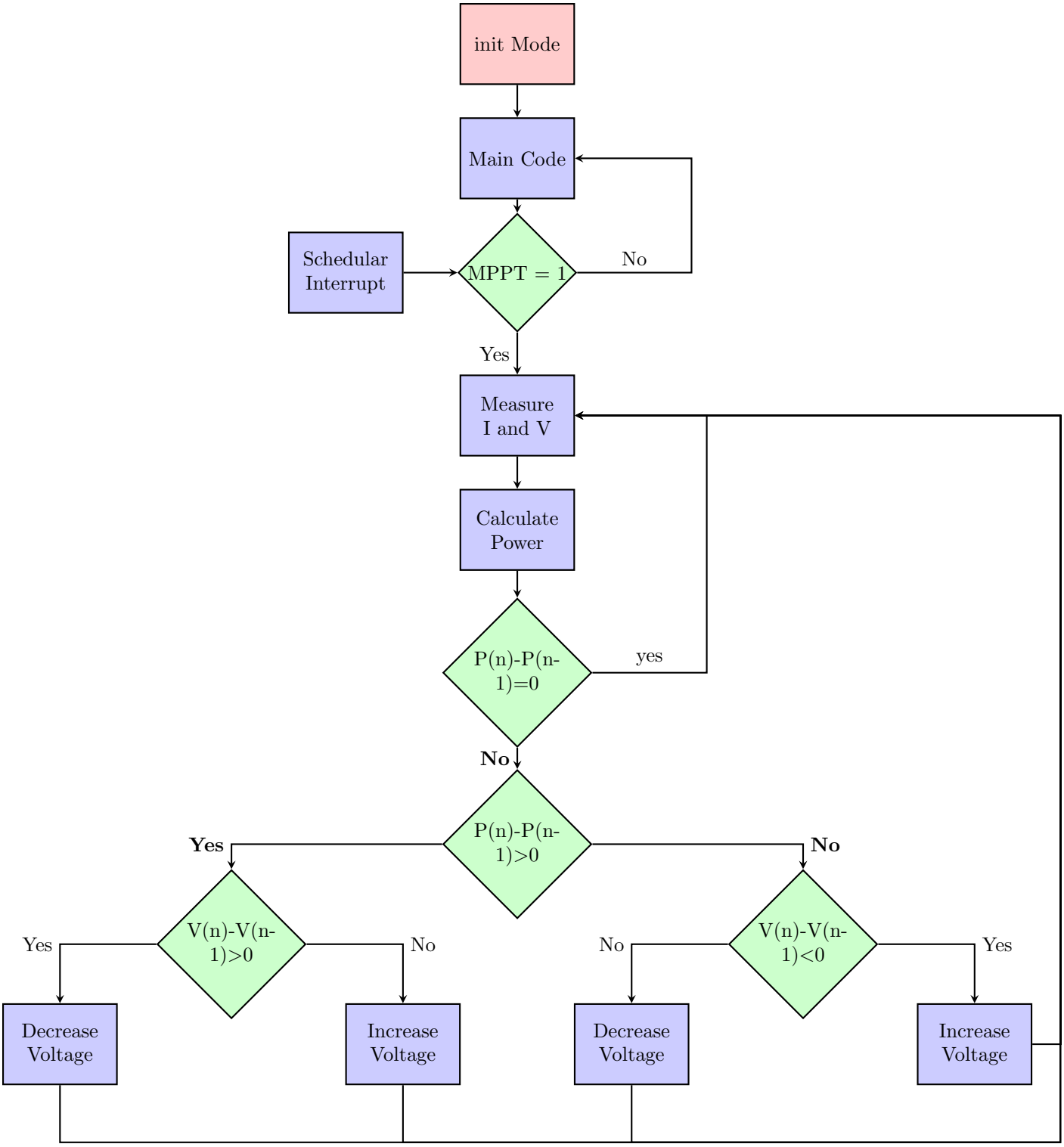

Fig 6 : Maximum Power Point Tracking

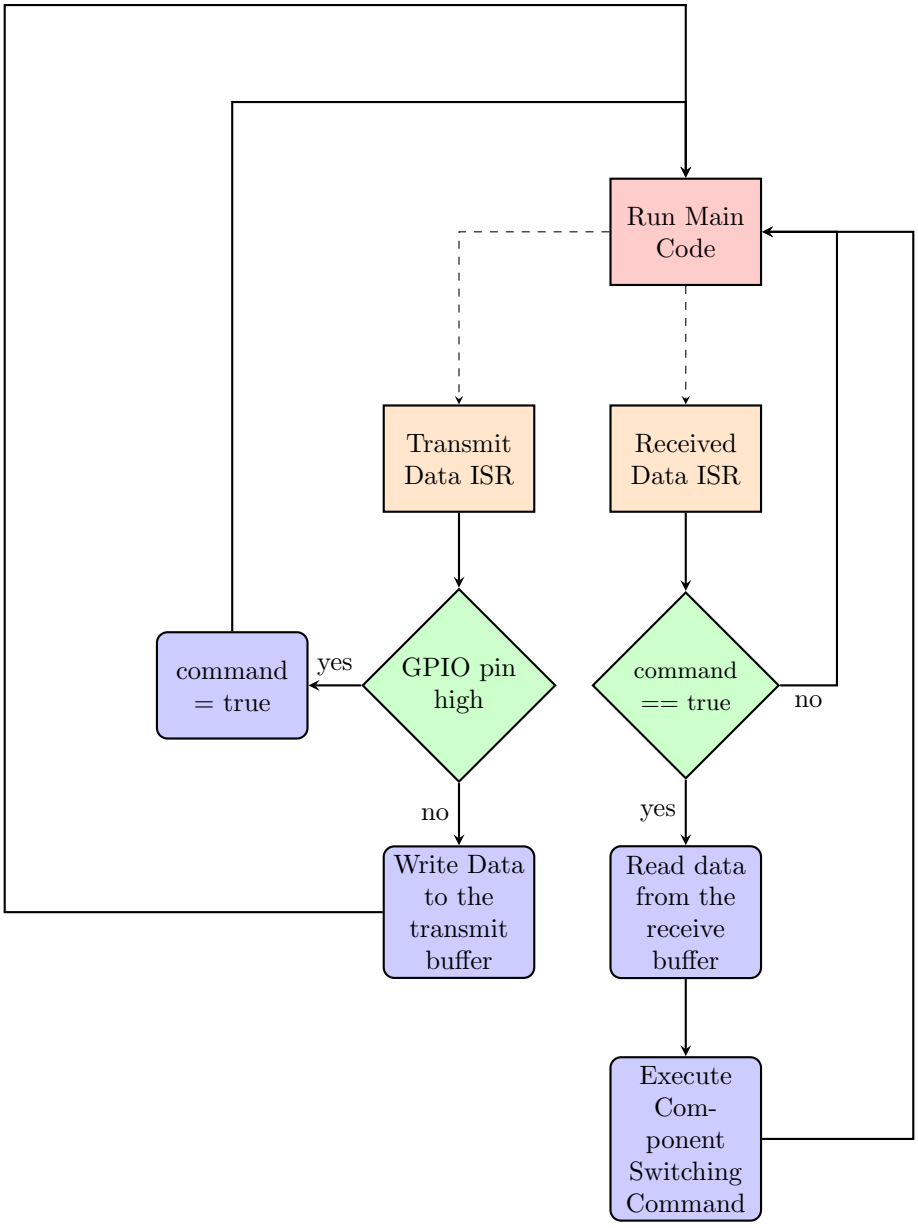

Fig 7 : On-Board Computer SPI Interfacing Task