Wake-Up Radio System for Batteryless IoT System

DESIGN DOCUMENT

sdmay21-14

Henry Duwe

Jacob Bernardi, Edmund Duan, Douglas Zuercher, Kwanghum Park, Bryce Staver, Zacharias Komodromos

sdmay21-14@iastate.edu

https://sdmay21-14.sd.ece.iastate.edu

Executive Summary

Summary of Requirements

The below list provides the requirements relevant to this project.

- Batteryless IoT Device with Wake-Up System
- Wakes up to an RF signal trigger
- Capable of node-to-node communication for signal relaying
- Low power consumption

Development Standards & Practices Used

In accordance with IEEE standards and good practices for electronic systems, our device will contain a well-organized schematic. In conjugation with multiple figures explaining each module of our project in detail. For this project we will need to design a printed circuit board or PCB, which follows basic rules of design such as those in the design rule check (DRC). Furthermore we need to follow radio frequency standards for any such traces on our PCB, as well as our transceiver which will operate at approximately 433 MHz. This includes a solid ground plane on the bottom layer of our board, proper on board shielding to protect other circuits, and in our case Keysight ADS simulations using FR4 substrate parameters to ensure our traces are sized correctly for maximum power transfer. Lastly for hardware, possibly noisy elements such as our switch mode power supply will be kept away from sensitive portions of our system.

Software will also be a sizable portion of our system design for programming the MCU. Proper documentation and in code comments will be included. This will follow standard practices for our chosen language and environment used, as well as memory used and limiting thereof.

IEEE standards relevant to project:

IEEE 610.7-1995 - IEEE Standard Glossary of Computer Networking Terminology

ISO 3309 - Information Processing Systems -- Data communication high level data link control procedure -- Frame structure

Applicable Courses from Iowa State University Curriculum

A number of courses taught at Iowa State University are applicable to this project. These courses include EE 230, EE 321, and EE 414. Courses such as EE 333, which are project-oriented, are also applicable.

New Skills & Knowledge Not Taught in Course Work

- PCB layout, footprint creation, and schematic drafting in Altium Designer
- Simulation of RF traces using Keysight ADS
- Specing components for use in designs
- RF harvesting methods and systems
- RF circuits and components (e.g. antennas)
- Signal transmission and modulation techniques
- Communication with companies
- Remote communication and collaboration skills among the team

Table of Contents

1 Introduction **6**
1.1 Acknowledgement 6
1.2 Problem and Project Statement 6
1.3 Operational Environment 7
1.4 Requirements 7
1.5 Intended Users and Uses 8
1.6 Assumptions and Limitations 8
1.7 Expected End Product and Deliverables 9

Project Plan **9**
2.1 Task Decomposition 9
2.2 Risks And Risk Management/Mitigation 9
2.3 Project Proposed Milestones, Metrics, and Evaluation Criteria 10
2.4 Project Timeline/Schedule 11
2.5 Project Tracking Procedures 11
2.6 Personnel Effort Requirements 11
2.7 Other Resource Requirements 12
2.8 Financial Requirements 13

3 Design **13**
3.1 Previous Work And Literature 13
3.2 Design Thinking 13
3.3 Proposed Design 14
3.4 Technology Considerations 16
3.5 Design Analysis 17
3.6 Development Process 19
3.7 Design Plan 19

4 Testing **20**
4.1 Unit Testing 20
4.2 Interface Testing 21
4.3 Acceptance Testing 21
4.4 Results 22

5 Implementation **22**

6 Closing Material **22**
6.1 Conclusion 22
6.2 References 23
6.3 Appendices 23

List of figures/tables/symbols/definitions

Figure 1: Propose Design State Diagram 7
Figure 2: Use-case diagram for harvester circuit 8
Figure 3: Task Decomposition for Project Design and Testing 9
Figure 4: Project Timeline for Fall 2020-Spring 2021 11
Figure 5: Decomposition of project tasks and effort requirements 12
Figure 6: Design Thinking Diagram 14
Figure 7: Proposed Design 16
Figure 8: Design Option 2 - Rejected 18
Figure 9: Design Option 3 - Current 18
Figure 10: Modified version of waterfall model 19
Figure 11: Powercast Harvester Test Results 22

1 Introduction

1.1 Acknowledgements

The senior design team would like to formally acknowledge and thank Henry Duwe, Assistant Professor at Iowa State University, for his technical advice on wake-up radios and low-power IoT devices as well as the use of his lab space and components. The team would also like to thank Vishak Narayanan, one of Professor Duwe's graduate students, for his assistance in testing and providing access to prior research.

1.2 Problem and Project Statement

Many Internet of Things devices are designed to be batteryless. In order for them to operate, some make use of an energy harvesting system. Once power is harvested, it is stored in a capacitor, which is then used to power the device with values neither constant nor of high magnitude. The result is then, when dealing with a batteryless device, it is highly likely, if not guaranteed, that there will be times when the system is not powered. In order to help with power usage, many devices poll and cycle on and off in order to extend the effective on-time. However, for a device that may need to be on when called upon, this presents the chance of missing the trigger. Therefore, a method of ensuring that the device is able to be triggered at any time while also maximizing the effective on-time requires implementation.

The solution presented by our team to this problem is the creation of a device which utilizes a wake-up system to wait for a wake-up signal to be received. Upon reception, the wake-up system will interrupt the microcontroller (MCU) to wake it up from a low power mode to normal operation. In doing so, we end up with a system where a user is able to instruct the MCU to follow a process, and it will enter a sleep mode when done. If it needs to activate at a certain time, it can do so by itself, but if the user wants an external stimulus, they can use the wake up trigger to accomplish this. This will create a more power efficient system that wakes up the IoT device. It allows for greater ability for IoT devices to be synchronized by the trigger signal, that is repeated by a device when received, and to perform tasks in unison if they have enough charge to be powered on and are in sleep-mode.

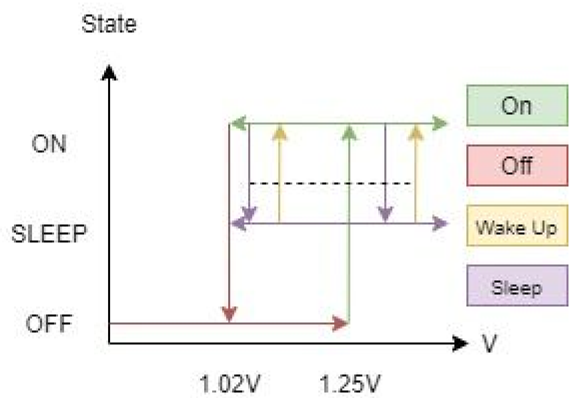


Figure 1: Propose Design State Diagram

1.3 Operational Environment

The end-product is expected to be operated in indoor commercial environments, such as office buildings and lab spaces, where electro-magnetic noise is not of significant concern and where RF signals are prominent. The environment is expected to be clean and climate-controlled such that dust and humidity are unlikely to interfere with electronics.

1.4 Requirements

The functional are for this system are as follows:

- The system should reject incorrect trigger signals, activating only when a valid trigger has been received
- The system will repeat the trigger signal once received
- When a trigger is received the device should enter normal operation
- The maximum average off time of a node should be no greater than one minute if a source is within 1m

The nonfunctional are for this system are as follows:

- The size of any node in the complete system should be no greater than 3” x 3”
- Each node should be batteryless, instead storing collected energy on a capacitor
- Nodes must be able to send and receive a trigger signal distances of at least two meters
- Nodes should operate correctly in temperatures above 0°C and below 50°C.

1.5 Intended Users and Uses

The product is intended to be used as a research medium to investigate batteryless, wireless RF communication between multiple IoT nodes. Professor Duwe and his graduate students are the intended users of this system. The biggest constraint that drives the project is the communication between two nodes, and specifically the ability for one node to trigger others. If a device receives the trigger, it will repeat it. The device's microprocessor will be able to run any arbitrary script the user intends. Once the script ends, the MCU will enter sleep mode at which point it can only exit sleep mode if the user programs a real-time clock event or the trigger signal is received. If the device does not have sufficient charge to be on, then it will not be able to trigger on.

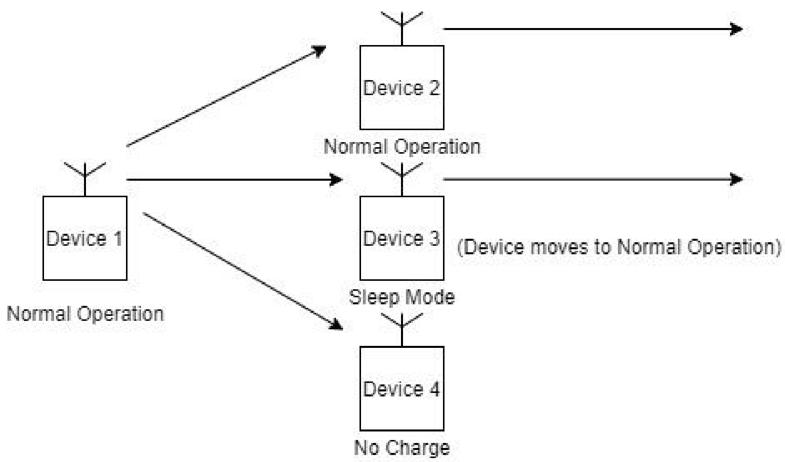


Figure 2: Use-case diagram for harvester circuit

1.6 Assumptions and Limitations

The validity of the proposed design is contingent upon a number of assumptions. These assumptions are listed below.

- Users will not configure MCU in a way which will make it consume more than 60mW, otherwise damage to the device may occur.
- Users will properly place the on-board MCU into standby, using provided documentation, when computations are not being performed
- Barriers which may impede transmission will not be placed between neighboring nodes
- The communication between the nodes will be a wireless RF signal

The proposed design is subject to several known limitations, listed below.

- Nodes may not be properly triggered when placed more than two meters away from neighboring nodes
- Nodes will not be resistant to high-noise environments
- Nodes will not be resistant to physical disruptions, such as shaking
- Nodes will not be provided with an ESD rating

- Nodes will not be designed to handle high-levels of humidity, dust, dramatic temperature swings, or temperatures outside of the designed operating region

1.7 Expected End Product and Deliverables

The end-product is expected to be a batteryless IoT device composed of a digital computing platform (a low-power MCU), supporting electronics for the MCU, an RF Harvester, and a Wake-Up system. The user should be able to use the onboard MCU for any purpose they need. The device should enter sleep mode if no user defined functions are running. Upon receiving an RF trigger signal, the device should exit sleep mode and repeat the trigger. In addition to the end-product, the team will deliver a demonstration which proves the functionality of the system.

2 Project Plan

2.1 Task Decomposition

The main tasks for this project are detailed in the figure below.

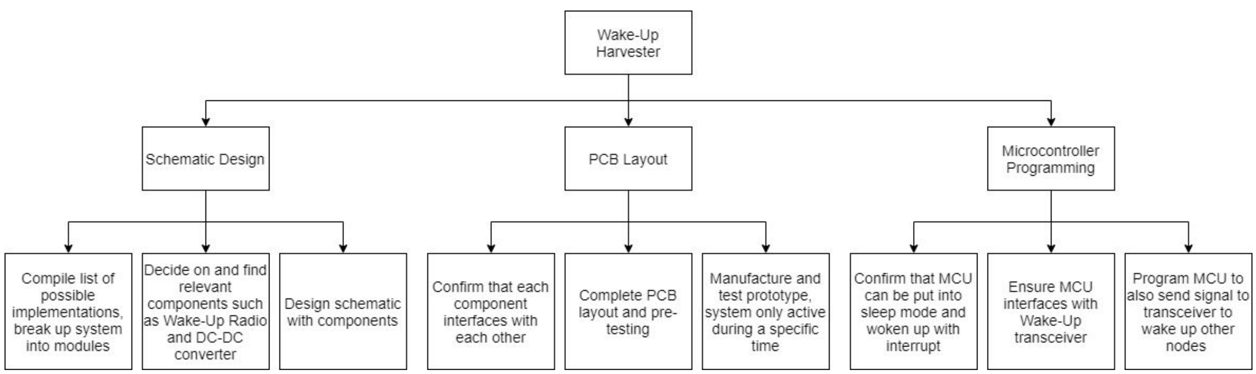


Figure 3: Task Decomposition for Project Design and Testing

2.2 Risks And Risk Management/Mitigation

The biggest risk associated with this project involves the system used to power the wake-up module, with a risk factor probability of around 0.3. The power source for the module will be a DC-DC boost converter that exclusively powers the wake-up module that takes a very low input voltage and boosts it up to a voltage that the module can operate at. The risk lies in the efficiency of the system since DC-DC boost converters are less efficient at low input voltages and low power applications. The team's method of minimizing risk is using a pre-made DC-DC boost converter which is made to be used for this purpose.

Another risk is the wake-up module itself, which is essentially a receiver and filter (risk factor probability of 0.2). While it is feasible for the team to make a custom module, it is very likely that it would add to the power consumption of the device, which is detrimental as it is necessary to keep power usage minimal. Beyond that, creating a high frequency filter with a precise enough band may prove expensive if attempted. For this reason, the team has opted for a RF wake-up IC that has these features built in and is optimized for low power consumption. In doing so, the team can almost guarantee the signal will be received and processed correctly and efficiently.

2.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

Milestone 1: A design concept that takes all requirements and assumptions into consideration with all initial bases covered on paper. Concept is peer-reviewed and approved by all team members.

Milestone 2: Creation of a low-level, detailed schematic for each module with generic parts. Simulations indicate that the system functions as intended.

Milestone 3: All components are chosen to spec. To pass this milestone, all devices necessary from the past milestone should be chosen to fit the design requirements. Based on guaranteed component specifications, the complete design should fall in a reasonable power consumption range. All devices which have SPICE models should be included in the previous milestones' simulation, and simulations should indicate that the system functions correctly.

Milestone 4: Completion of PCB layout and routing, with design rule checks passed. Peer-review and approval are necessary for the milestone to be completed.

Milestone 5: PCB ordered, received, and populated. This milestone is complete when the board has been fully populated and is ready for testing.

Milestone 6: Testing of revision one is complete and changes necessary to ensure the proper operation of revision two have been identified.

Milestone 7: Revision two schematic revisions and simulations complete, along with board layout and routing.

Milestone 8: Second revision PCB ordered, received, and populated. This milestone is complete when the board has been fully populated and is ready for testing.

Milestone 9: Final revision tested and confirmed to be functional per required specifications.

2.4 Project Timeline/Schedule

WBS NUMBER	TASK TITLE	START DATE	DUE DATE	COMPLETE
1	Revision 1			
1.1	List of possible implementations	S1 W1	S1 W8	90%
1.1.1	High-level system design	S1 W8	S1 W11	20%
1.2	Low-level schematic complete	S1 W11	S1 W14	10%
1.3	Component selection	S1 W12	S1 W14	10%
1.4	Testing and schematic confirmation	Interim W1	Interim W2	0%
1.5	PCB Layout and routing	Interim W2	Interim W6	0%
1.6	Testing (rev. 1)	Interim W7	Iterim W8	0%
2	Revision 2			
2.1	Schematic updates	Interim W8	S2 W1	0%
2.2	Testing and schematic confirmation	S2 W2	S2 W4	0%
2.3	PCB Layout and routing	S2 W4	S2 W6	0%
2.4	Testing (rev. 2)	S2 W6	S2 W7	0%
3	Final Revision			
3.1	Schematic updates	S2 W7	S2 W9	0%
3.2	Testing and schematic confirmation	S2 W9	S2 W10	0%
3.2.1	PCB Layout and routing	S2 W10	S2 W11	0%
3.2.2	Testing (final)	S2 W11	S2 W13	0%

Figure 4: Project Timeline for Fall 2020-Spring 2021

2.5 Project Tracking Procedures

Our team is using Git, Google Drive, Trello and Discord in order to track the progress of our project. Git will be used later in the process for version control and maintenance of our hardware design files (PCB and schematic implementations). This tool is being used because it provides an easy way to have version control and to have a centralized place where all members of the team will be able to pull the current working versions from. Google Drive is used to store agendas, meeting notes, images, as well as any collaborative assignments done in EE 491. It provides the team with in-browser tools to be able to complete these documents collaboratively while working in a virtual environment. Trello is a centralized place to keep notes, documentation and links to all of our devices. It is also a method of implementing SCRUM since we can use its tools to organize ourselves both time and action-wise. Finally, Discord is used for quick communication and sharing general thoughts regarding meetings and the overall project plan. It provides a way for us to organize different aspects into separate channels, join calls and share our screens through a simple process.

2.6 Personnel Effort Requirements

Task	Number of people	Number of hours required (cumulative among members)
Understand what the project is about, how the device for project works and what components are	6 members	20 hours

in the device		
Research on technologies to use and list the options to choose the best one by comparing the pros and cons	6 members	40 hours
Split up the components to research the wake-up system	3 members	20 hours
Work on Rev. 1 (Design, assembly and test)	6 members	70 hours
Work on Rev. 2 (Design, assembly and test)	6 members	40 hours
Work on final rev. (Design, assembly and test)	6 members	20 hours
Weekly meeting with advisor and client to report the project status and plan	6 members	1 hour per week
Weekly group meeting to discuss about the project and check if everyone is on the right track	6 members	2-4 hours per week
Write a Bi-weekly report to update the project status and explain about the plan for next week	6 members	1.5 hours per week

Figure 5: Decomposition of project tasks and effort requirements

2.7 Other Resource Requirements

To complete this project we will first and foremost need access to a PCB Design software. In our case, we will be using Altium Designer, which is provided by the university. In addition to this tool, we will need a custom PCB designed by the team, which can be purchased and received within 2 weeks. In addition to the board, we will need a variety of IC components, some of which include a Powercast RF harvester, a Wake-Up IC, an MCU, antennas and a few surrounding components that we will populate the board with. Lastly because we are taking in an RF signal we need to design our traces correctly and for this we are using Keysight's Advanced Design System to size our traces correctly for maximum power transfer.

2.8 Financial Requirements

The different components used in this project will not exceed $150. On top of this, we will need to pay for the PCB itself for which the price can vary from vendor to vendor. A good estimate for the PCB is around $200.

3 Design

3.1 Previous Work And Literature

Although we are the first senior design team to take on a Wake-Up harvester, the concept of Wake-Up radios has existed before, despite being relatively recent [1]. Wake-Up radios are described as devices that are active only when they receive a specific signal or trigger and many have been made in the past few years [1]. In our system, we are using a premade component for our wake-up radio in addition to premade components for our power harvester, MCU, and DC-DC regulator. The datasheets for these components are listed in section 6.2.

3.2 Design Thinking

When defining the scope of this project, we initially believed that we would have to modify the harvester itself. However, in the “define” stage, it became more apparent that the solution would not be that straightforward. Seeing as though we would not have access to the controller on board the harvester to activate the harvester’s output, we moved to a few other designs before settling on our proposed solution. After defining the project, we then moved to the “ideate” phase, in which we began looking for low-power components that could work in the system such as a Wake-Up radio and a DC-DC regulator. The components we could find drove our design, since there is a small market for low enough power components, which meant our design was fairly constrained. Due to these constraints, we opted for a transceiver instead of a Wake Up Radio marketed IC.

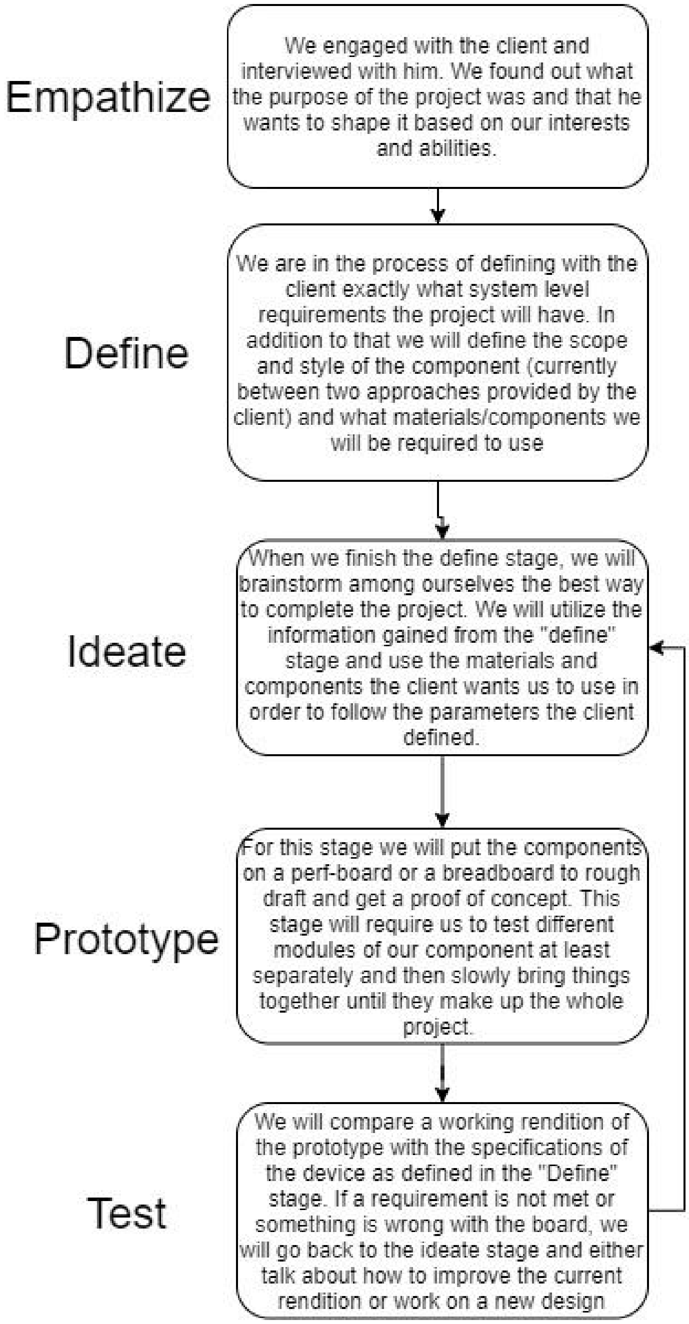


Figure 6: Design Thinking Diagram

3.3 Proposed Design

A number of solutions to the problem of creating IoT devices that can perform computational tasks while operating on a low-power or inconsistent power supply already exist, with all solutions focusing on ways in which power can be conserved when not needed. The most popular way of conserving power is by placing the computing device, typically some form of a microcontroller, in a sleep state until an internal watchdog timer triggers and reactivates the device. In this fashion, the

MCU can be set to use minimal power for the vast majority of the time, only consuming significant amounts of power during the intervals where it activates and performs computational tasks.

While this solution is undoubtedly efficient, it is only appropriate for IoT devices that do not need to network with other devices or are networking with systems that are always active (for example, an IoT device that periodically downloads data from a server). For devices that need to communicate periodically with other periodic devices, this solution is not suitable since it depends on devices re-activating at precisely the same times. For small meshes, this may be achievable by occasionally ensuring that both devices are properly synchronized, but the reliability of this solution decreases dramatically for meshes containing more than a trivial number of nodes.

It is clear that a more appropriate solution is a system that shifts the responsibility of reactivating the MCU from the MCU's internal watchdog timer to a low-power device capable of networking with other devices, a wake-up radio or always-on low-power radio transceiver being examples. With these devices, the MCU can be configured to only re-activate when a set of packets on a designated carrier frequency are received, allowing for device activation to occur on the receiving of a signal external to the node itself. In theory, this also introduces the ability for a node to wake up neighboring nodes by reconfiguring its transceiver to transmit packets, allowing the opportunity for the mesh size to not be limited by the range of the original transmitter.

Shifting the responsibility of waking the MCU from the MCU itself to another component creates an additional opportunity, being the opportunity to *completely* unpower the MCU when not in use. This saves additional power since the MCU is no longer even maintaining basic bare-bones operations, and in theory helps get some of the power back that is being spent to maintain the operation of the always-on transceiver. This optimization was heavily explored by the team, but ultimately the decision was made to not use this power-saving strategy for one primary reason: while more efficient, it significantly reduces the number of tasks that can be performed by the system. For example, the user may decide that they want the MCU to still reactivate automatically after a period of time, whether or not it received an external trigger, so that it can complete another portion of its task. Completely unpowering the MCU removes this ability, limiting the system's uses. It was thus decided to not unpower the device and to instead continue just placing the device into a sleep mode.

The final design proposed by the team is a design that incorporates three primary components: an RF harvester, a low-power always-on transceiver, and a low-power MCU. The RF harvester will harvest and store energy from background RF signals on a capacitor, which will then be used by the harvester's in-chip boost regulator to power the MCU. The capacitor will also interface with an additional low-power high-efficiency boost regulator, which will power the always-on transceiver and necessary circuitry for the transceiver.

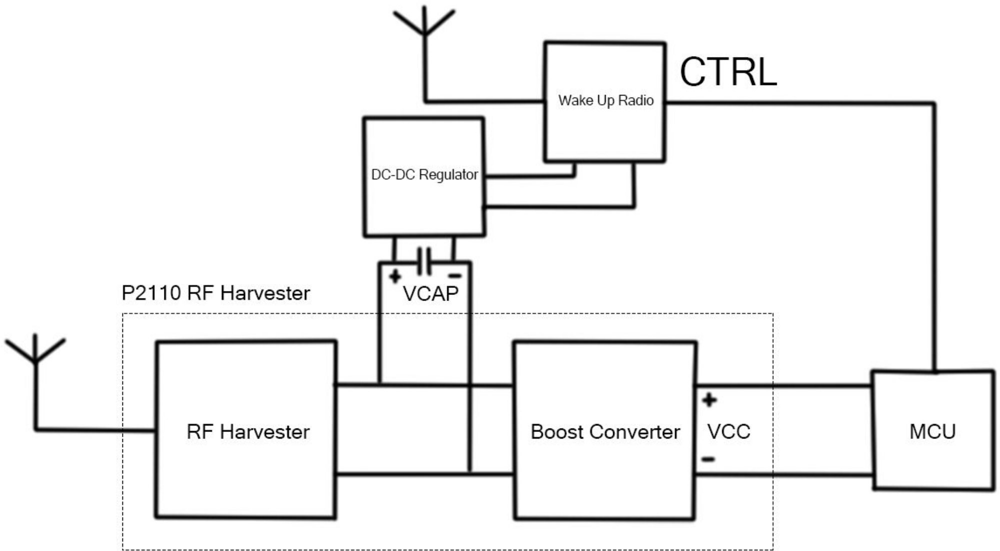


Figure 7: Proposed Design

The transceiver and MCU will be connected in such a way that the two devices can communicate with each other through SPI, allowing the MCU to reconfigure the transceiver at will and allowing for the transceiver to wake the MCU. During normal operation, the MCU will be in a sleep state where only functions such as interrupts handlers and watchdog timers are active. The transceiver will be in a wake-on radio mode, waking up at intervals to briefly check if a carrier signal is present. When a carrier signal is present and a valid packet of data is detected, the transceiver will trigger the MCU's external interrupt, waking the device from sleep mode and activating it. The MCU will complete its tasks and re-enter sleep mode.

3.4 Technology Considerations

The greatest technological consideration that should be accounted for in the system is the power consumption of each node. Because each node is a device that harvests its own power, there are two effective limitations on power supply: how much the nodes can harvest in a given period and how much the nodes will consume in a period. The necessity for the nodes to be wireless means that these challenges are unavoidable.

To consider the topic of how much the nodes can harvest in a given period of time, one must first look at the harvester that the customer encouraged the team to use, as it is the source of the power used by each node. Unfortunately, the harvester in question, the Powercast P2110B, does not provide detailed information on its typical harvesting capacity and capabilities. It instead refers to stating its maximum capacity and capabilities. As a result of this, additional testing is needed before harvesting considerations can be made.

While the consideration of how much the nodes will harvest in a period is defined by the capabilities of the harvester being used, the measurement of how much the nodes will consume in a period of time is not. This evaluation depends on the power consumption of the components in each node and on the size of the capacitor we decide to use. The larger the capacitor, the more storage capacity the node will have and thus the longer the nodes can keep their on-board MCUs active. The downside is that it will take longer to charge and thus may keep the system out of commission for longer. On the other hand, a smaller capacitor will charge quickly but won't be able to support the system for long, especially on high power operations.

To mitigate the issue, and maximize the energy storage while decreasing the off time, our team needed to choose very low power components and try to decrease consumption as much as possible. During the design phase we had a decision to make on how to power the trigger receiver module of the device. It could either be powered from the Powercast harvester or the capacitor itself. Using the option of powering off the Powercast as a base case, we looked into methods of improving the consumption on paper. This led to the solution of using an external DC-DC converter that had a higher efficiency at low voltages compared to the Powercast in order to power the trigger receiver module.

In concurrence with minimizing consumption, the MCU module of the device has transmission capabilities. However, its capabilities are limited to a small number of transmission protocols. This prompted us to explore other options, and in doing so we found transmitters that would operate with lower consumption rates compared to the MCU when transmitting. Having a need for both transmission and receiving, we could now spec a transceiving device as our trigger module that would act either as a receiver or as a transmitter with a higher efficiency than the base case of using the MCU, as well as not adding to the total number of components used.

3.5 Design Analysis

Each iteration of the design process had an issue that significantly changed the overall design. Starting from the first design, we assumed there was a way to bypass the harvester module's hysteresis loop to force it on during its off cycle. While it looks like a good plan in theory, after looking into the datasheet and trying to get in touch with the manufacturer, we concluded that it was not possible to have access to the on board DC-DC converter to trigger it on. Instead of making our own harvester in which we could control the DC-DC converter powering the rest of the devices, we opted to stay with the 3rd party harvester since our knowledge is limited and implementing a harvester would significantly add complexity to the project that was unnecessary.

For the second design idea, its simplicity is what was most attractive. The biggest issue with this design is that the MCU would not be off when the trigger is received. Specifically, since the onboard harvester's hysteresis loop is not something we have access to, the trigger won't change the operation if the MCU is active. On the other hand, if the MCU is off, it is because the Powercast DC-DC converter is off, meaning that it is in the charging state. If the trigger connected the MCU to the Powercast output, it would still not turn on. Thus this design was redundant and didn't meet the goals and requirements of the final device.

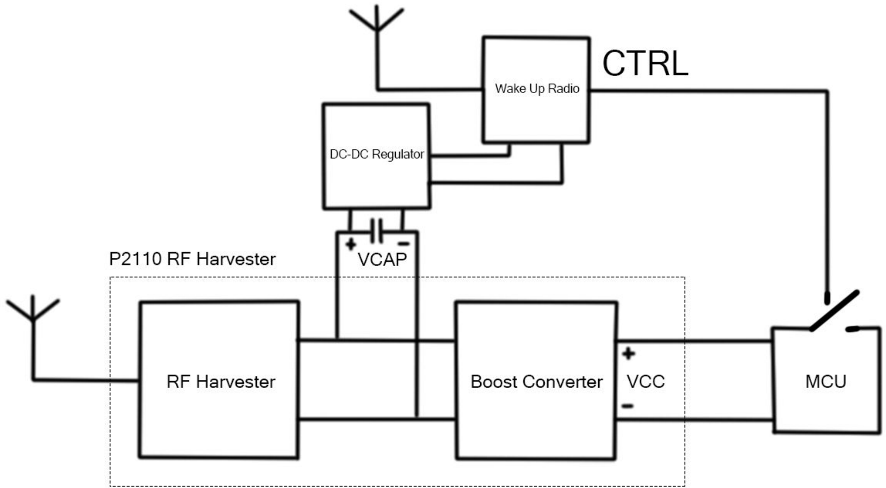


Figure 8: Design Option 2 - Rejected

Finally, the current design is to make use of the MCU's sleep mode. With this method, the trigger would take the MCU out of sleep mode when triggered. What this ends up achieving is, if the user has intensive operations running on the MCU which require it to be on, it will operate the device as usual. On the other hand, if the device is idle, instead of consuming power while not accomplishing anything, it can go into sleep mode. If the user has time sensitive operations that are internal to the device, the MCU is fully capable of getting itself back to normal operation. On the other hand, if the user has time sensitive operations that don't rely on an individual device, they can use an external wake-up signal to trigger the MCU on. What this ultimately achieves is a longer lifecycle for the device, and a way of externally triggering it on.

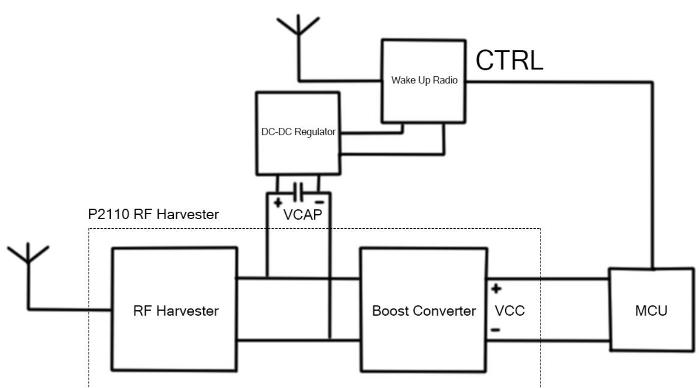


Figure 9: Design Option 3 - Current

3.6 Development Process

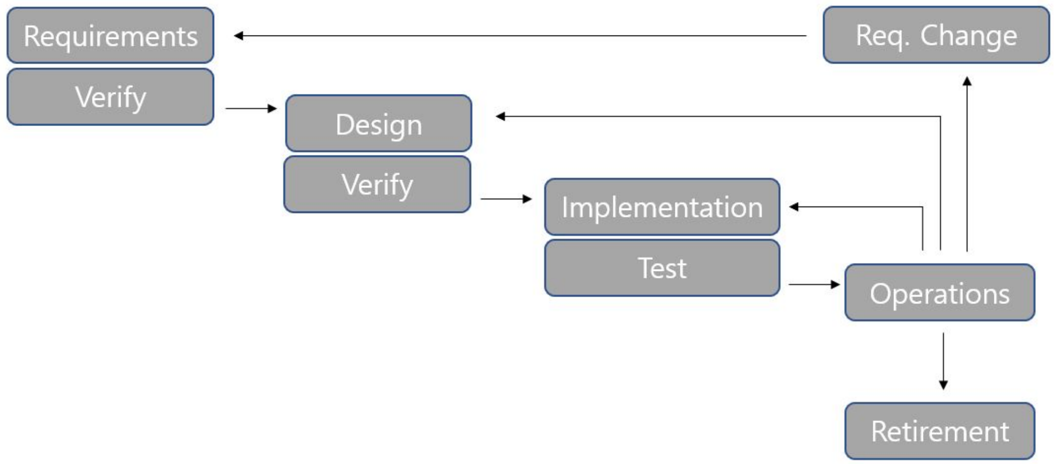


Figure 10: Modified version of waterfall model

We decided to choose the development called “Modified version of waterfall model.” The original waterfall model has the disadvantage of not allowing repetition. Therefore, it is inflexible and monolithic to the requirements. We have to design the high-level systems like MCU, Wake-up radio, and DC-DC regulator, and we are required to change the design of them whenever there is a problem with testing or verifying processes. For our project, we have to combine all these parts together to make sure they are working well together. Because of this, we have to test and change it often. However, the original waterfall model is not flexible, because it cannot proceed to the next stage until the previous stage is completed. To solve this disadvantage, we chose the modified model. This model has more flexible features for documentation and for analysis of changing requirements.

Also, when we get access to our components and have space for testing, we will move into a test-driven design process. The reason for this is we rely at first on on-paper material to get as many details of the project planned out, and as soon as we have access to work with materials in hand we will start using our results from module-oriented testing to optimize and refine the overall design. The benefit of this is we won’t have to backtrack as much in the processes if something fails, and it also provides the opportunity for a multi-front approach where failure or setbacks in one does not affect the others.

3.7 Design Plan

To design and build our project, we are going to use components available in the market. We first had to research for the reasonable Powercast RF harvester, a Wake-Up IC, an MCU, and DC-DC regulator. Our project is intended to be used by wireless IoT device manufacturers with low power consumption. Considering the requirements, we had to find the right components which can be run by low power.

For our design, we are going to use Altium Designer. Altium Designer is an electronic circuit design software that provides the entire process of PCB development such as circuits and simulations in an integrated development environment. The whole process of development can be developed efficiently because it can be carried out through a single program.

We set the workspace to design the footprint and high-level system of components to test if our chosen components work. Our project needs to be verified and tested, because one part may not be operated well by another and we need to make sure that all the components are connected and work well.

4 Testing

4.1 Unit Testing

The main parts of this project that will be tested in isolation will include the RF harvester, the wake-up transceiver, and the MCU. Much of this testing will be done using a power supply so that we can take our time debugging and not rely on the intermittent power the RF harvester would provide. To start we will be testing the RF harvester on its own so that we can ensure that it can take in an RF input and produce a boosted DC output. In addition to ensuring that it operates correctly, this will be an opportunity to understand what amount of power we can expect to collect when this device is in an office setting. This information will be valuable in confirming that the components selected will be viable with this power source.

Another component that will be unit tested is the wake-up radio system that utilizes a sub-GHz frequency transceiver to send and receive a “wake-up signal.” The purpose of testing this device on its own is to confirm that the power drawn is less than the average power that the power harvested will be. If the transceiver draws less than the average power harvested, we can be confident that the system will be able to receive and send a “wake-up signal” at any given time and that the harvester will not limit its functionality.

Lastly, we will be testing the MCU’s functionality in isolation before combining it with the rest of the system. The main things that we want to test include making sure that we can send a “wake-up” interrupt to the MCU and that it can put itself into sleep mode when not performing computations. The interrupt will be coming from the sub-GHz transceiver but before interfacing the two systems we will make sure that our MCU program can recognize an interrupt and leave sleep mode to complete its task. Note: The task that the MCU performs is not in the scope of this project. The last thing that the MCU needs to do is once that task is completed it should take the proper steps to safely re-enter sleep mode.

For a more detailed explanation of how we will interface and configure each device, please refer to the Appendices.

4.2 Interface Testing

Once the proper unit tests have been completed, we will be ready to begin interfacing the MCU, RF Harvester, and Wake-up Radio system to work together as an RF Harvesting IoT device. One of the most important interface tests will be making sure that the Wake-Up radio system can send and receive the “wake-up signal.” This means that we will be using two separate radio systems where one will act as the transmitter and one as the receiver. The purpose of this is to ensure our custom “wake-up signal” can be transmitted and received between devices, which was a main feature desired by the client. Related to this test, we will also need to check that once the wake-up signal has been received, the transceiver can send another signal to the MCU that will act as the interrupt to leave the sleep state.

The other important interface test is to confirm that the RF power harvester can support the wake-up radio system at any given time. This would mean making sure that the wake-up radio system isn’t intermittently turning on and off because it constantly drains the capacitor the harvester is connected to. Assuming that the harvester is able to support the wake-up radio, the other thing to test would be that the harvester’s boost converter can turn on and supply 3.3V to the MCU. If there are no issues supplying the necessary power to the other components, we can be confident that the RF harvester will be able to consistently support the entire device.

4.3 Acceptance Testing

To demonstrate that the design requirements for the final product are all met, an overall test of the system will be performed. Multiple nodes will be set up throughout a room such that each periodically turns on and performs a simple task before entering a sleep mode which will be exited automatically after some delay on the devices’ internal watchdog timers. The purpose of this is to demonstrate that the MCU can activate itself periodically, a requirement that customers may need. Therefore, driving an LED to be active for a few seconds can be a sufficient method to display that this criteria has been accounted for. The nodes will be allowed to run for several hours to ensure that they are deactivating and not extraneously draining the charge available to each node. Throughout the experiment (near the beginning, middle, and end), the nodes will be triggered by the primary transmitter. If the nodes are operating as expected, they will all simultaneously activate and perform their task. Doing this multiple times throughout the lifetime of the experiment will prove that the nodes are fully functional even after running for a long duration of time and there has been enough time for the charges on each node to deplete dramatically.

Once confirming that all nodes are able to wake-up from the central transmitter, the experiment will then be repeated, but with the nodes spaced farther and not all located around the transmitter. If nodes outside the range of the transmitter are able to activate upon the transmitter being activated, this will prove that the nodes are capable of forwarding the wake-up signal to other neighboring nodes.

4.4 Results

The capacitor on the Powercast harvester can ideally store 0.0391 J of energy and had a value of 50 mF. The load used was the CC1352R MCU. The maximum output current the regulator can reach is

50 mA. Since we are operating at 3.3V and the MCU draw is approximately 20mA max (not constant), for reasonable performance, the consumption of the load should be less than 60 mW.

	Minimum on time	Charging time	Time capacitor can hold charge	Maximum off time
.5 m from transmitter	.4395 sec	.25 sec	0.1895 sec	6.64 sec
1 m from transmitter	.6102 sec	.25 sec	.3602 sec	46.995 sec

Figure 11: Powercast Harvester Test Results

5 Implementation

For some time prior to and within the next semester, we plan on having each individual component tested to ensure that they operate within their specifications before creating schematics and assembling them on a PCB. Specific components that we will test would be the antenna for the transceiver and making sure that the MCU works well with the Wake Up radio. Specific details regarding our schedule are listed in the above Figure 2, where we will have our first circuit made during winter break and hopefully spend the semester testing and improving upon the design based on the results found. We will make changes to the schedule if necessary. For example, if we are able to have a functional second revision, we would not spend the next six weeks developing a third revision.

6 Closing Material

6.1 Conclusion

So far in the process we have focused on defining the goals of the project as well as deciding on different ways to accomplish that. In regards to the goals, our aim was to create a wireless and self-sufficient component that has processing capabilities in addition to transceiving capabilities such that it can be triggered and can trigger other components. Using these goals, we went through multiple ideas of implementation, until we settled on our current solution. Using an RF harvester, supporting the self sufficiency, to power an MCU we can fill a majority of the goals. Adding a transceiver also provides the capability to transmit and receive which allows for the triggering option. Finally, in order to keep power consumption low to encourage larger on-time due to the power source being the harvester, we decided to use a DC-DC converter in addition to the one on board on the harvester, in order to power the transceiver. Of all configurations considered, our final one was chosen mostly due to the constraints of the components that are available on the market as well as the points the client wanted us to hit. All of our other configurations failed the specifications in one way or another. After having decided on a design, we have moved forward to finalize selection of the components and are beginning to test them individually and modularly in order to get proof of concept as well as some hard data from our own testing rather than relying on datasheets which were under specific circumstances. Moving forward, we are finalizing a design in Altium PCB Designer to have one PCB board to house all these components which will be the final product of the project.

6.2 References

[1] "CC1352R," Texas Instruments, May-2020. [Online]. Available: https://www.ti.com/product/CC1352R. [Accessed: 23-Oct-2020].

[2] L. Wilhelmsson and D. Sundman, "Wake-Up Radio – A key component of IoT?," Ericsson.com, 18-Dec-2017. [Online]. Available: https://www.ericsson.com/en/blog/2017/12/wake-up-radio--a-key-component-of-iot. [Accessed: 23-Oct-2020].

[3] "Low Input Voltage Boost Converter," Texas Instruments, Dec-2014. [Online]. Available: https://www.ti.com/lit/ds/symlink/tps61202.pdf?HQS=TI-null-null-digikeymode-df-pf-null-wwe&ts=1600912252796. [Accessed: 23-Oct-2020].

[4] "P2110B Datasheet," Powercast, 2016. [Online]. Available: https://www.powercastco.com/wp-content/uploads/2016/12/P2110B-Datasheet-Rev-3.pdf. [Accessed: 23-Oct-2020].

6.3 Appendices

Powercast harvester testing:

The Powercast harvester was placed on a development board with an antenna and the CC1352 microcontroller. To find how it performed in our desired setting, the device was set at .5 and 1 m away from the transmitter in 310 Durham and was tested a total of five times.

Transceiver Testing:

The AX5043 can easily be programmed by connecting to any MCU through a register file accessed serially via SPI (Serial Peripheral Interface) protocol. It communicates by sending frames while in frame mode via a 256 Byte FIFO. The programming datasheet recommends also using the interrupt pin and SYSCLK line to share the AX5043 clock with the MCU for more precise data communication.

During transmission, only the write port is accessible by the microcontroller. During reception, only the read port is accessible by the microcontroller. Otherwise, both ports are accessible through the register file. Before the device can transmit, though, there are some fundamental characteristics that need to be chosen:

Parameter	Description
f_{XTAL}	Frequency of the connected crystal in Hz
modulation	FSK, MSK, OQPSK, 4−FSK or AFSK (for recommendations see below)
$f_{CARRIER}$	Carrier frequency (i.e. center frequency of the signal) in Hz
BITRATE	Desired bit rate in bit/s
h	Modulation index, determines the frequency deviation for FSK $32 > h \geq 0.5$ for FSK, 4−FSK or AFSK, $f_{deviation} = 0.5 * h *$ BITRATE h = 0.5 for MSK and OQPSK (For AFSK, $f_{deviation}$ is usually set according to the FM channel specification. For 25 kHz channels, it is often approximately 3 kHz)
encoding	Inversion, differential, manchester, scrambled, for recommendations see the description of the register ENCODING.

From the different forms of modulation, it is best to choose MSK, because it has higher sensitivity to FSK and a smaller preamble than PSK.

We can use AX_RadioLab to compute the register settings for the AX5043 for a friendlier and faster way of setting it up.

In order to transmit and receive, we can use the following flowcharts:

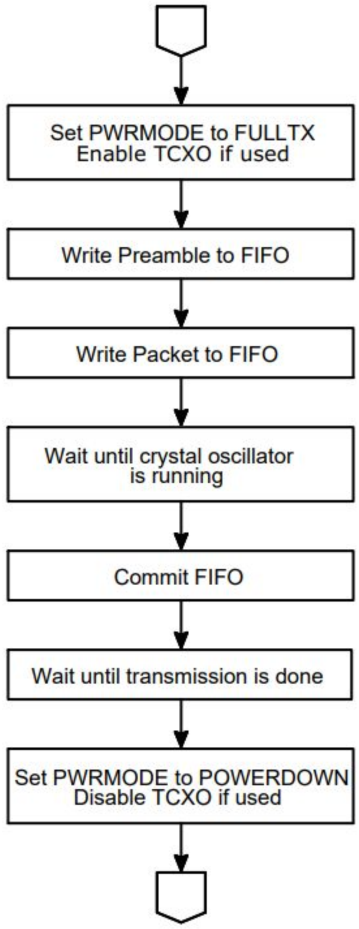
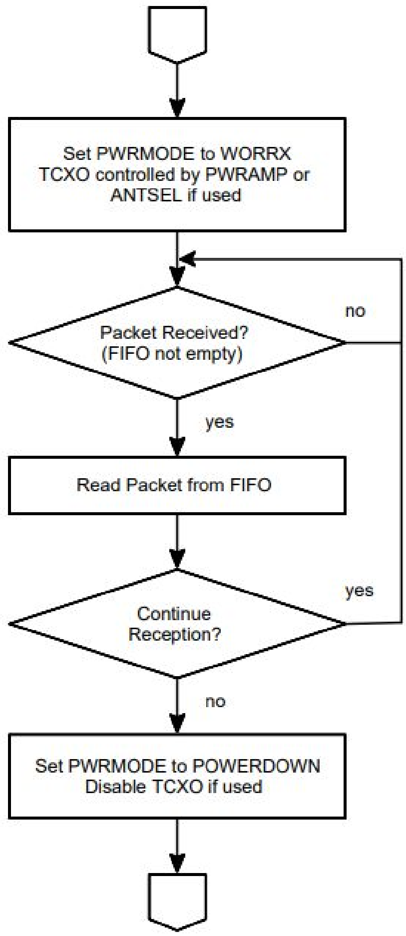


Figure 9. Transmitter Flow Chart



MCU Testing:

The CC1352 MCU will be tested using development boards and programmed to confirm that it can be put into a low-power “sleep mode” and woken by an interrupt. Using Code Composer Studio we will be able to modify the necessary registers to configure the microcontroller to enter and exit sleep mode when necessary. The following appendix describes what registers are modified to achieve this.

SCR Register: Power-Management functionality

Table 2-138. SCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
4	SEVONPEND	R/W	0h	Send Event on Pending bit: 0: Only enabled interrupts or events can wakeup the processor, disabled interrupts are excluded 1: Enabled events and all interrupts, including disabled interrupts, can wakeup the processor. When an event or interrupt enters pending state, the event signal wakes up the processor from WFE. If the processor is not waiting for an event, the event is registered and affects the next WFE. The processor also wakes up on execution of an SEV instruction.
3	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.
2	SLEEPDEEP	R/W	0h	Controls whether the processor uses sleep or deep sleep as its low power mode 0h = Sleep 1h = Deep sleep
1	SLEEPONEXIT	R/W	0h	Sleep on exit when returning from Handler mode to Thread mode. Enables interrupt driven applications to avoid returning to empty main application. 0: Do not sleep when returning to thread mode 1: Sleep on ISR exit
0	RESERVED	R/W	0h	Software should not rely on the value of a reserved. Writing any other value than the reset value may result in undefined behavior.

We will be setting the following bits in this register:

- Bit 4: (0) We only want the processor to “wake-up” on our trigger
- Bit 2: (1) We want the processor to use deep sleep to conserve maximum power