2 Project Plan

2.1 Project Management/Tracking Procedures

Which of agile, waterfall or waterfall+agile project management style are you adopting? Justify it with respect to the project goals.

Agile, agile will allow us to go with the flow and continue to iterate and refine the project as it grows. This methodology makes more sense for us for now as a large chunk of this project is understanding the systems and standards in play. It is difficult to define exactly what work needs to be done without having a strong understanding of the systems as a whole. As our understanding of the system(s) grow, our goals and mission will also.

What will your group use to track progress throughout the course of this and the next semester. This could include Git, Github, Trello, Slack or any other tools helpful in project management.

Discord for general communication and collaboration. Issues/milestones will be tracked in gitlab as well as weekly meeting minutes.

2.2 Task Decomposition

In order to solve the problem at hand, it helps to decompose it into multiple tasks and subtasks and to understand interdependence among tasks. This step might be useful even if you adopt agile methodology. If you are agile, you can also provide a linear progression of completed requirements aligned with your sprints for the entire project.

- **Prepare Rasp Pi with OS and Python libraries**
    a. **Python 3.8 (may need to change)**
    b. **Py Sunspec 2**
- **Raspberry Pi is usable as a TMS microcontroller**
    a. **Develop a master controller arbitration algorithm**
    b. **Add OMG DDS library functionality to Rasp Pi**
    c. **Create command issuing and receiver software**
- **Microcontroller can accept user input to determine configuration, and rollout to the TMS network**
    a. **Determine capabilities of the inverter and network to be exposed to users**
    b. **Design a user interface/input system**
    c. **Convert inputs to commands on the TMS network/local device**
- **Raspberry Pi can interpret TMS commands and issue SunSpec commands to configure microgrid pallet inverters**
    a. **Test scripts with SunSpec python library**
    b. **Create configuration sequences for various setups**

2.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

What are some key milestones in your proposed project? It may be helpful to develop these milestones for each task and subtask from 2.2. How do you measure progress on a given task? These metrics, preferably quantifiable, should be developed for each task. The milestones should be stated in terms of these metrics: Machine learning algorithm XYZ will classify with 80% accuracy; the pattern recognition logic on FPGA will recognize a pattern every 1 ms (at 1K patterns/sec throughput). ML accuracy target might go up to 90% from 80%.

In an agile development process, these milestones can be refined with successive iterations/sprints (perhaps a subset of your requirements applicable to those sprints).

- **Prepare Raspberry Pi software**
- **Communication between two Raspberry Pis**
- **First test communication between two microgrids**
- **Reliable master/slave determination**
- **Fully TMS compliant**
- **Fully SunSpec compliant**
- **Full stable, autonomous communication between two microgrids**

2.4 Project Timeline/Schedule

• A realistic, well-planned schedule is an essential component of every well-planned project

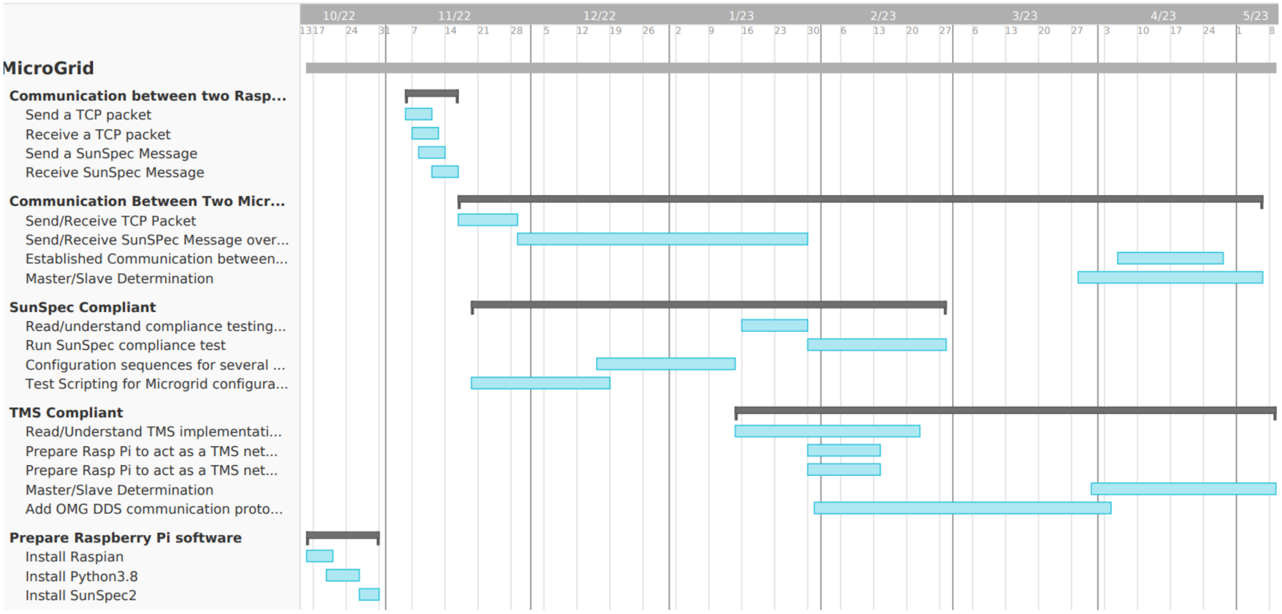
• Most scheduling errors occur as the result of either not properly identifying all of the necessary activities (tasks and/or subtasks) or not properly estimating the amount of effort required to correctly complete the activity

• A detailed schedule is needed as a part of the plan:

– Start with a Gantt chart showing the tasks (that you developed in 2.2) and associated subtasks versus the proposed project calendar (including both 491 and 492 semesters). The Gantt chart shall be referenced and summarized in the text.

– Annotate the Gantt chart with when each project deliverable will be delivered

• Project schedule/Gantt chart can be adapted to Agile or Waterfall development models. For agile, a sprint schedule with specific technical milestones/requirements/targets will work.



2.5 Risks And Risk Management/Mitigation

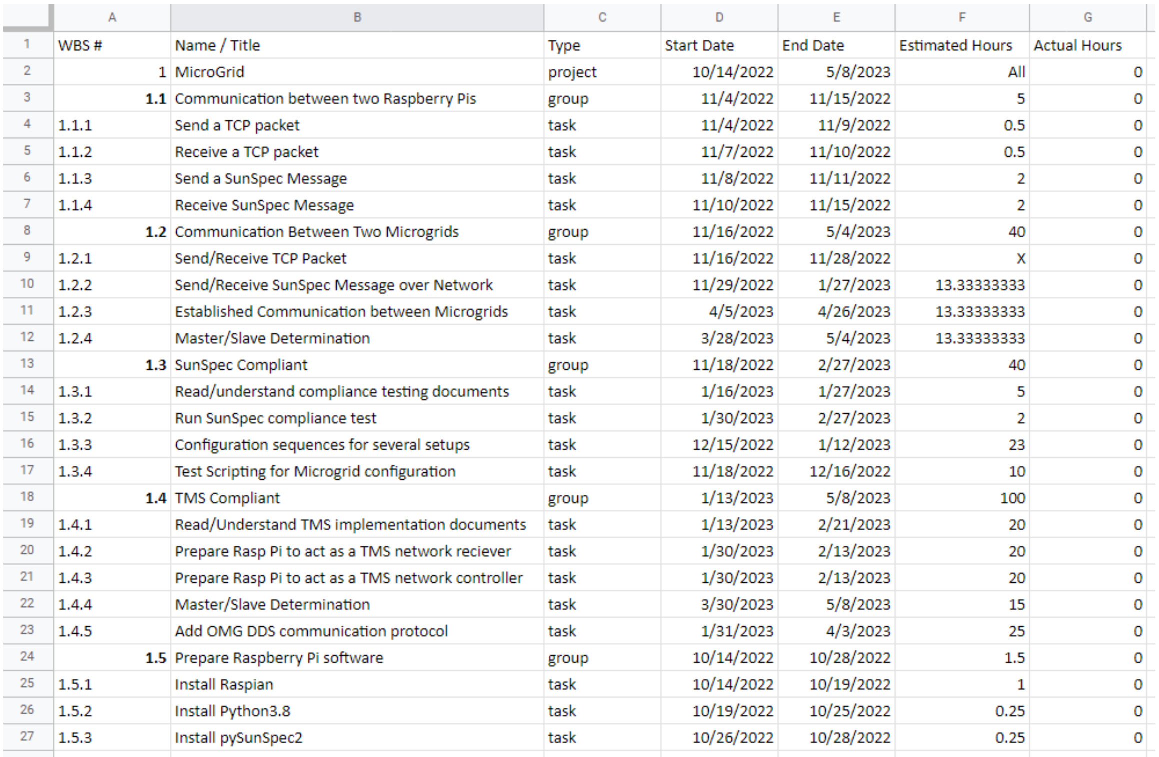
Consider for each task what risks exist (certain performance targets may not be met; certain tools may not work as expected) and assign an educated guess of probability for that risk. For any risk factor with a probability exceeding 0.5, develop a risk mitigation plan. Can you eliminate that task and add another task or set of tasks that might cost more? Can you buy something off-the-shelf from the market to achieve that functionality? Can you try an alternative tool, technology, algorithm, or board?

Agile projects can associate risks and risk mitigation with each sprint.

- **There is a risk of damaging the microgrid units. The microgrid**
- **There may be security flaws in the way we handle communication between units as we are not very experienced in networking or network security.**

2.6 Personnel Effort Requirements

Include a detailed estimate in the form of a table accompanied by a textual reference and explanation. This estimate shall be done on a task-by-task basis and should be the projected effort in the total number of person-hours required to perform the task.



	A	B	C	D	E	F	G
1	WBS #	Name / Title	Type	Start Date	End Date	Estimated Hours	Actual Hours
2	1	MicroGrid	project	10/14/2022	5/8/2023	All	0
3	**1.1**	Communication between two Raspberry Pis	group	11/4/2022	11/15/2022	5	0
4	1.1.1	Send a TCP packet	task	11/4/2022	11/9/2022	0.5	0
5	1.1.2	Receive a TCP packet	task	11/7/2022	11/10/2022	0.5	0
6	1.1.3	Send a SunSpec Message	task	11/8/2022	11/11/2022	2	0
7	1.1.4	Receive SunSpec Message	task	11/10/2022	11/15/2022	2	0
8	**1.2**	Communication Between Two Microgrids	group	11/16/2022	5/4/2023	40	0
9	1.2.1	Send/Receive TCP Packet	task	11/16/2022	11/28/2022	X	0
10	1.2.2	Send/Receive SunSpec Message over Network	task	11/29/2022	1/27/2023	13.33333333	0
11	1.2.3	Established Communication between Microgrids	task	4/5/2023	4/26/2023	13.33333333	0
12	1.2.4	Master/Slave Determination	task	3/28/2023	5/4/2023	13.33333333	0
13	**1.3**	SunSpec Compliant	group	11/18/2022	2/27/2023	40	0
14	1.3.1	Read/understand compliance testing documents	task	1/16/2023	1/27/2023	5	0
15	1.3.2	Run SunSpec compliance test	task	1/30/2023	2/27/2023	2	0
16	1.3.3	Configuration sequences for several setups	task	12/15/2022	1/12/2023	23	0
17	1.3.4	Test Scripting for Microgrid configuration	task	11/18/2022	12/16/2022	10	0
18	**1.4**	TMS Compliant	group	1/13/2023	5/8/2023	100	0
19	1.4.1	Read/Understand TMS implementation documents	task	1/13/2023	2/21/2023	20	0
20	1.4.2	Prepare Rasp Pi to act as a TMS network reciever	task	1/30/2023	2/13/2023	20	0
21	1.4.3	Prepare Rasp Pi to act as a TMS network controller	task	1/30/2023	2/13/2023	20	0
22	1.4.4	Master/Slave Determination	task	3/30/2023	5/8/2023	15	0
23	1.4.5	Add OMG DDS communication protocol	task	1/31/2023	4/3/2023	25	0
24	**1.5**	Prepare Raspberry Pi software	group	10/14/2022	10/28/2022	1.5	0
25	1.5.1	Install Raspian	task	10/14/2022	10/19/2022	1	0
26	1.5.2	Install Python3.8	task	10/19/2022	10/25/2022	0.25	0
27	1.5.3	Install pySunSpec2	task	10/26/2022	10/28/2022	0.25	0

1.1 Communication between two Raspberry Pis: The goal here is to simulate the end product with less moving parts. Sending and receiving TCP packets should be a simple websocket connection. In order to send a SunSpec message there may be more involved with using the SunSpec api to do so.

1.2 Communication Between Two Microgrids: To accomplish this task, we need to learn more about microgrids what they will be exchanging in their communication. This seems like a task that will be much more involved than a simple TCP message going from one microcontroller to another. We are also factoring in the time it will take to get acquainted with the pySunSpec library as well.

1.3 SunSpec Compliant: Most time spent here will be understanding the tests and how they need to be run to test compliance. The running of the tests themselves should not be overly tedious. Depending on results of the test, there may need to be adjustments made to the configurations. Some time spent on understanding initial configurations may show up here also.

1.4 TMS Compliant: The Tactical Microgrid Standard doesn't seem to have a straightforward or automated way to test for compliance. Compliance testing here will most likely require a high level of understanding of TMS and other related standards, including Sunspec and OMG DDS protocols. In this process the team will also have to work out Master/Slave assignment in an automated manner. This on its own may be a heft goal.

1.5 Prepare Raspberry Pi software: This process should be pretty straightforward, the longest part may be simply downloading the OS image onto the pi.

2.7 Other Resource Requirements

Identify the other resources aside from financial (such as parts and materials) required to complete the project.

- **Semi-Regular access to micro-grid pallets**
- **Raspberry Pi's (2x)**
- **Documentation on API's and associated tech**
  - **Microgrids**
  - **Sunspec**
  - **Tactical Microgrid Standard**
  - **OMG DDS**