Master Branch | Project Backlog

This is the project backlog for the **Master Branch** team in CS361 Spring 2018.

The rows are sorted by the sprint (i.e. sprint 0-4). There are corresponding tasks, stories, solutions, estimated time, and due dates for each ToDo.

Tasks are assigned to team members by listing initails (or ALL) inside of the task cell, and are colored based on their level of priority (see priority key).

Team Members: Matt McFadden (MM), Nic Cox (NC), Jingtao Yang (JT), Ben Yuhas (BY), Zach Kennow (ZK)

Sprint/ Priority	Tasks	Story	Solution	Estimated Time	Due	Priority key
Sprint_0:				(Hr)		High
1	Submit Lab3 to D2L. (ALL)				2/12 @ 5: 00pm	Medium
2	Create initial project backlog (sprint0) and start populating it with scenarios, tasks, etc. (MM)			2	2/12 @ 5: 00pm	Low
3	Come up with at least 3 use cases and their description, as well as create a use case template. (BY)			1	2/12 @ 5: 00pm	done
4	Come up with at leat 5 questions of clarification. (NC)			1	2/12 @ 5: 00pm	
5	Think about use cases & make appropriate diagrams/descriptions using the use case template. (ALL)					
6	Investigate the domain of sports timing and add stories to the prject backlog. (JT)					
Sprint_1:						
1	Create a simulator that issues events for the Chronotimer and has a system clock. (NC)	The client would like to test the software of the system without the hardware components.	The client can use the simulator to test the software.	2	3/1 @ 11: 00am	
2	Implement the console or file input option. (NC)	The client would like to test the software by inputting commands into the simulator via the command line and a file.	The client can use the simulator and choose between console and file input.	1	3/1 @ 11: 00am	

3	Implement the EXIT command for the simulator (NIC).	How does a user exit the simulated enviroment for the timing system?	The user types/uses the EXIT command, and the system exits gracefully making sure to store/reset any necessary properties/data.	1	3/1 @ 11:00am	
4	Create a UML diagram of ChronoTimer package (MM).		UML.pdf	2	3/1 @ 11:00am	
5	Create ChronoTimer package and associated classes according to UML(see release 1.0) (ALL)	A client would like to have a deliverable product of the ChronoTimer 1009 as soon as possible.	Version 1.0 of the timing system	15	3/1 @ 11:00am	
6	Implement the Run class (BY).			1	3/1 @ 11:00am	
7	Implement the Racer class (BY).			1	3/1 @ 11:00am	
8	Implement the Controller class (MM).			2	3/1 @ 11:00am	
9	Implement the Clock class (BY).			1	3/1 @ 11:00am	
10	Implement the Channel class (NC).			1	3/1 @ 11:00am	
11	Implement the Parser class (NC).			1	3/1 @ 11:00am	
12	Implement the Racer class (BY).			1	3/1 @ 11:00am	
13	Implement the Printer class (MM).			1	3/1 @ 11:00am	
14	Design/create quiescent state (ALL).	What state is the system in after power up?	The system will be in the quiescent state.		3/1 @ 11:00am	
15	Implement/think of design for quiescent state (ALL).	What level of functionality is required while in the quiescent state?	No races active, ready to receive commands.	2	3/1 @ 11:00am	
16	Implement the POWER on and off methods for Controller (MM).	A user would like to turn the system on/off?	Use the POWER command. A squence of shut down procedures such as clearing any buffers, closing files, etc. for powering off, and likewise for powering on.	1	3/1 @ 11:00am	

17	Implement IND event adn newRun method for Controller (MM).	A user would like to time an individual race with a specified start and finish.	The user can choose the type of race to be individual (IND) (default type for sprint1).	1	3/1 @ 11:00am	
18	Implement the RESET method for Controller (MM).	Can another event be recorded if there is a current event?	No, to start another event there must be no other events currently. A user can restart/reset the timer with the RESET command.	1	3/1 @ 11:00am	
19	Implement the TIME command, and setTime method for Clock and Contoller (BY).	How does a user set the internal clock/timer of the system?	If the user want to set the internal clock of the system, which by default is handled by the firmware, they can use the TIME <hour>:<min>:<sec> command to set the exact time of the system.	1	3/1 @ 11:00am	
20	Implement the TOG method for Controller and Channel (MM & NC).	How can a user activate/toggle a channel on/off?	The user can use the TOG <num> command where <num> is the associated channel they wish to toggle.	1	3/1 @ 11:00am	
21	Implement the CONN method for Controller and Channel (MM & NC).	How can a user connect a sensor to a channel?	Use the CONN <sensor> <num> command where num is the channel to connect the sensor to.	1	3/1 @ 11:00am	
22	Implement TRIG mthod for Controller (MM).	A user wants to start a channel (timer) on the system.	The user can use the TRIG <num> command where <num> is the channel the wish to trigger.	1	3/1 @ 11:00am	
23	Implement the queues for the Run class as well as the addRacer method (BY).	How does the system keep track of all the racers?	The system keeps a queue of the racers in the correct order of first to last.	1	3/1 @ 11:00am	
24	Implement the SWAP method for Conroller and Run (MM).	What happens if the racer in second place passes the racer in first?	Use the SWAP command, which swaps the two lead racers.	1	3/1 @ 11:00am	
25	Implement the PRINT method for Controller, Run, and Racer (MM & BY).	How does the user see the results of a run?	Use the PRINT command to print the results of the current race.	1	3/1 @ 11:00am	
26	Implement the DNF method for Run, Racer, and Controller (MM & BY).	What happens if a racer does not finish?	The user must use the DNF <num> command to record that the racer did not finish (DNF).	1	3/1 @ 11:00am	
27	Implement the CANCEL method for Run and Controller(MM & BY).	A sensor fails to trigger and the start/finish time is not recorded.	In this case there may be a fault in the sensor/system and further investigation/maintainance may be required. The race may need to be canceled and restarted.	1	3/1 @ 11:00am	
28	Implement the CANCEL method for Run and Controller (BY).	A racer has a false start and the race needs to be restarted (ie the timer reset).	The user can use the CANCEL command to terminate the current race and start over.	1	3/1 @ 11:00am	

29	Implement the START method for Controller, Run, and Racer(MM & BY).	How can a user easily start an individual race (ie toggle chennel 1)?	The user can use the START command after channel 1 is plugged in and toggled to the on state.	1	3/1 @ 11:00am	
30	Implemet the FINISH method for Controller, Run, and Racer (MM & BY).	How can a user easily end an individual race (ie toggle chennel 2)?	The user can use the FINISH command after channel 2 is plugged in and toggled to the on state.	1	3/1 @ 11:00am	
31	Handle/implement error condition: time > 9,999.99s.	The time for a race exceeds 9,999.99 seconds.	An error condition is met and an exception is thrown.	1	3/1 @ 11:00am	
32	Write test files for the system/simulator (ZK).	How can a developer test the TRIG and TOG commands?	There will by system tests (Junit tests) that test each command.	2	3/1 @ 11:00am	
Sprint_2:						
1	implement exportation of Run to a text file in Json format upon the EXPORT<RUN> command. (Luis) [1]	How can a user capture the data from a Run?	When the EXPORT<RUN> command is executed all relevant data for the current run will be exported to a file named RUN<RUN> where <RUN> is the # of the run.	1	3/16 @ 11:00PM	
2	Merge/Integrate new team members (Andy & Luis) as well as establish whos implementation/code we are going to use for the rest of the project.		Sprint2.zip		3/16 @ 11:00PM	
3	Implement the PARIND race type as well as fix known bugs and other issues with the system (Andy).	How can a User record a Parallel Individual race type? That is how can the system simulate two parallel IND races at the same time?	The User may specify PARIND as the event type prior to the NEWRUN command. After that they may add racer to the Race via the NUM command implemented in sprint1.	3	3/16 @ 11:00PM	
4	Create a brief test plan as well as test cases/files to test the current iteration of the system (Matt).	How can a developer/client know if the system works properly?	There will be a test plan implemented - although the test plan for Sprint2 will be brief, further updates will be made to ensure complete method coverage, component unit tests etc. (see link ->)	Test Plan	3/16 @ 11:00PM	
5	Update the UML model (Zach).			1	3/16 @ 11:00PM	
6	Finish all of the current Use Cases for this sprint (Ben & Nic).		Use Cases	2	3/16 @ 11:00PM	
	Limit racers names as "001" - "999".	What are the input specifications for bib numbers? i.e. should a Bib number of "001" be displayed exactly like that or is if okay to display "1"?	The bib number of a racer will be in the range 000-999, so racer "1" would be displayed as "001".			

	Implement the CLEAR method for Controller and Run.	How can a user clear a particular racer from a run?	Use the CLEAR <num> command where num is the racer that is to be cleared.			
	Implement the disconnect method (DISC) for the Channel and Controller Class.	How can a user disconnect a sensor from a channel?	Use the DISC <num> command where num is the channel that is to be disconnnected.			
	Add an outer class/wrapper class for Run which allows for multple individual "Runs" simulating PARIND.					
	Implement an interface/display feature that shows a sensor is connected to a channel.	How will a user know that a sensor is connected to the timing system?	A display feature in the simulator will show that a sensor of type {EYE, GATE, PAD} is connected to channel <NUM>.			
	Integrate the TRIG command with a button on the console (GUI).	A user wants to manually start a channel (timer) on the system.	The user can push a button on the console that correstonds to the channel effectivily starting the timer.			
	Implement the restriction of adding the same racer (i.e. racers w/ same name) to the same run.	Can two racers with the same name/number be in the same run/heat?	No, racers with the same name/run cannot be apart of the same run/heat.			
	Add a feature to the reset method that writes any relevent data (e.g. current race times, etc.) into a file for backup.	When the RESET command is used, will it delete all existing times and other data, or will that data be stored elsewhere?	Any data will be lost, unless it has been saved (not implemented in first release).			

[1] There is an issue/bug with the Gson API that needs to be sorted out later due to the time constraint... Basically Gson won't convert our Race class to Json format.