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), Andy Gasparini(AG), Zach Kennow (ZK), Luis Gonzalez (LG)

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)	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	
SPRINT_3						
1	Fix all of the bugs with Sprint2 (i.e. EXPORT & PRINT)			2	3/30 @ 12:00PM	

2	Create GUI for the simulator and integrate with current version. (AG)	A user would like to be able to use commands through a graphic interface.	The user may use an implemented GUI that represents commands as graphic buttons and settings. The user may execute a desired command by activating the command's respective button or selection.	2	4/19 @ 5:00PM	
3	Implement a number pad in the GUI for entering numbers for commands. (AG)	A user would like to be able to use the GUI to enter numbers instead of typing them in.	The user may enter the numbers 0 - 9 into the program by activating their respective button.	2	4/19 @ 5:00PM	
4	Add GRP event type to Run class. (MM)	A user would like to time a race with a group of racers running in parallel with the same start time.	The user may specify a race type GRP prior to the NEWRUN command and add racers using NUM implemented in Sprint 1. Racer numbers not required, they are returned after the command FINISH as places 00001, 00002, 00003, for 1st, 2nd, and 3rd place, respectively.	3	4/19 @ 5:00PM	
5	Allow a channel without a paired sensor to be triggered. (MM)			1	4/19 @ 5:00PM	
6	Implement an interface/display feature that shows a sensor is connected to a channel. (AG)	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>.	1	4/19 @ 5:00PM	
7	Implement an interface feature for channels, clicking on one channel will attach (CONN command) a type of sensor selected in a drop down list and show the button as selected. Clicking again disconnects the sensor (DISC). (AG)	How will a user connect and disconnect a sensor from a channel?	The user will be able to click on a channel and select a sensor type from a dropdown menu. clicking it again will disconnnect the sensor.	1		
8	Add a Lane class (i.e. a composite class for Run). (MM)			1	4/19 @ 5:00PM	
9	Integrate the TRIG command for specific channels with a button on the console. (AG)	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.	1	4/19 @ 5:00PM	
10	Unit testing for Run & Race Classes. (NC, AG)			2	4/12 @ 5:00PM	
11	Continue testing for Clock & ClockInterface Classes. (NC, AG MM)			2	4/12 @ 5:00PM	

12	Implement the CLEAR method (CLR <Num>) for RaceEventsManager and Pool. (MM)	How can a user clear a particular racer?	Use the CLEAR <num> command where num is the racer that is to be cleared.	1	4/19 @ 5:00PM	
13	Implement the disconnect method (DISC <Num>), which takes a channel ID as a parameter. (AG)	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.	1	4/19 @ 5:00PM	
14	Add the RESET Button. (AG)	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 by Export.	1	4/19 @ 5:00PM	
15	Implement Swap function for IND type Run. (MM)	If a racer passes another, how can the user change the FINISH command to act on the new leading racer's channel? (IND only)	Use the SWAP command to switch the leading two racers, making the FINISH command act on the new lead racer's second channel. (IND only)	1	4/19 @ 5:00PM	
16	Limit racers names as "001" - "999" and do not allow for Racer's to have the same bib numbers. (AG)	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".	1	4/19 @ 5:00PM	
17	Do the UML model (ZK, LG)			1	4/15 @ 9:00PM	
18	Do the backlog (ZK, MM)			1	3/30 @ 8:00PM	
19	Do Use Cases (ZK, BY, LG)			1	4/15 @ 9:00PM	
20	Do more automated file testing (MM, AG, NC)			1	4/15 @ 9:00PM	

	Do GRP code model (AG, MM)	How does using the GRP command work within the system?	Using the event type GRP will initialize a Run with a race id, event type of GRP, and lane with two channels activated (i.e. channels 1 & 2). The command START will move all Racers in the Pool to the active queue for Lane 1, and trigger CH1, starting the Clock and setting each Racer's timeStartFormatted to the current system time. Each use of the FINISH command will put a Racer into the record queue of Race, and set the Racer's finish time, as well as record their place for this GRP Run (i.e. 00001, 00002, etc.). When the Run is printed instead of printing the Racer's bib numbers they will be ID'ed as 00001, 00002, 00003, and so on, corresponding to which Racer was finished first, second, third, etc.	1	4/25 @ 9: 00PM	
21	Add a running display to the GUI	The user would like to be able to see printed information as well as entered information in real time.	The GUI will have a running display of information such as entered numbers, Racer information, start/finish times, and general Run information.	1	4/19 @ 5: 00PM	
22	Add a Printer to the GUI	The user would like a Printer display of events as they happen	The GUI will have a running Printer display that shows events happening in real time (same as what is printed to the console for previous sprints).	1	4/19 @ 5: 00PM	
SPRINT_4						
1	Add PARGRP event type (AG)	The user would like to record a Parallel group event (i.e. a run with up to 8 racers each having their own finish channel).	The user can choose PARGRP event before starting a newrun	1	5/9 @ 5: 00PM	
2	Test PARGRP event (AG, MM)			1	5/9 @ 5: 00PM	
3	Create document for non-functional requirements (look at lab14)			1	5/3 @ 5: 00PM	
4	Create division of labor/participation document (look at lab14)			1	5/3 @ 5: 00PM	
5	Prep for presentation (ALL)			2	5/8 @ 1: 00PM	
6	Document code (ALL)			1	5/9 @ 5: 00PM	

7	Update the UML Model (LG, ZK)			1	5/9 @ 5:00PM	
8	Update Test Plan (MM, AG, NC)			1	5/9 @ 5:00PM	
9	Update Use Cases (LG, ZK, NC)			1	5/9 @ 5:00PM	
10	Make config.txt & racers.txt for server/displayresults (MM)			1	5/9 @ 5:00PM	
11	Create Server for displaying Runs (Use Lab 10/11 as model) (NC, ZK)	Spectators would like to view the results of a run without looking at the ChronoTimer.	The spectator can visit a specificied URL on the network of the ChronoTimer.	1	5/9 @ 5:00PM	
12	Export list of racers upon ENDRUN to Server (MM)	The Server needs to be sent the results of a run (racer's times).		1	5/9 @ 5:00PM	
13	Modify RaceEventsManager to send results to Server upon ENDRUN (MM)			1	5/9 @ 5:00PM	