CS 452 Project 2

Felix Fung (f2fung)
Dusan Zelembaba (dzelemba)

July 23, 2013

1 How To Run

```
> load -b 0x00218000 -h 10.15.167.4 "ARM/f2fung/p2_final.elf
> go
```

2 Submitted Files

Files listed here can be found under `/u1/f2fung/cs452/p2/`

2.1 md5sums

```
6b95cd7a797ed011c80bfd1c9c9f3a60  ./Makefile
8dd9b113c5d58039b523402022b5d182  ./debug.c
e6fe878a12d13ea01c7cdb38895772f8  ./context_switch.s
3db9d3f453cad4f61c79884d500379c9  ./reservation_server.c
1dc118c000601dc8accbed9bf54a2076  ./orex.ld
3a8b24b398ece870c61432b0cd3b3160  ./ourlib.c
3774f518f194d500c51de712c567e3d8  ./first_task.c
18e880c6ff334eb5f682c21dcbaf2c61  ./track_data.c
3b096de077660480045b6f57a090e482  ./icu.c
52b8990e115e43c17cb418b3eb2e8741  ./dijkstra.c
0044a9d5892b591929afcd304891ad7f  ./run_tests.c
cf871e452ca5cddbc3995304b57831ce  ./track_node.c
351eefdf131296b5c5ffdd324afabd01  ./track_edge_array.c
586458ee65db0afd0b1c4a39461b0795  ./main.map
820ae458f6e19c1fca45371dc7ccd2c7  ./ioserver.c
696a458ab20b11b20bb389abd145f753  ./location_server.c
feba44366f52d656b8811adba24da103  ./project.c
6a025c0ed584875d4bc92812ac0ae68c  ./sensor_server.c
92b17d4751cd6351ad5937f28952e828  ./train.c
c1705146beff0edb48142664c208d7fc  ./user_prompt.c
b9814306df3c3ce8fbf871b91a0121ae  ./nameserver.c
296f7f84cfb2c088fad7b6a69268d00b  ./data_structures/bitmask.c
```

```
fa8f8a366df3c90c69f6d4bdcc3e0a3a  ./data_structures/heap.c
b7891ab62d1d88925f871a99a4e047b9  ./data_structures/heapplus.c
89d74272f5d03eb54db03f8633cb2abb  ./data_structures/linked_array.c
6009877e08b2eb92fefe76ccdba82c27  ./data_structures/queue.c
5a913deb0a153b5cd55bb5c52f27215e  ./data_structures/simple_sm.c
d8e007eaadc6b69717224d34b4682969  ./test_helpers.c
792ad97b51eece81a637452da6056674  ./strings.c
6ef4c4581e33ad9c58ab73a896fc1cb2  ./rps_server.c
f351ccf69b83adb1e2aa0ef559a11eb8  ./include/all_tests.h
35bd0114e8113fbd214e3a389b4844c3  ./include/bitmask.h
4c86aadc2cca610cd67c7dd645371d33  ./include/clockserver.h
f14b1bd18405e4a13aa29074b4d3c265  ./include/context_switch.h
6234f8bf104c4eb3a280e512df891a36  ./include/debug.h
e68c6f9cbaee62d844dc83784b936ad3  ./include/dijkstra.h
9b124aeb5f84630af58c3abc2843c1a6  ./include/events.h
f5d5f01a856e51feeefb0c672539e687  ./include/first_task.h
e80eed468576da5be1cb04b8572681ae  ./include/heap.h
9d32c0d251a4f75f201ce8ebe295eb32  ./include/heapplus.h
60122cea48e43550bf96143b8310ab30  ./include/icu.h
574e859de18cdbe722764e3230f226ca  ./include/idle_task.h
c5b1026567dfc98fa4be22dde2163d98  ./include/interrupt_handler.h
bbd1a58e9662dc9efab4ffbeb4655ab0  ./include/ioserver.h
3ac7a3b2def34929bbd3198847a84c50  ./include/kernel.h
1f1dbb640125c1ea0cf60f07910134b6  ./include/linked_array.h
6cd532bf50de3b84920d11deb91c0ef9  ./include/location_server.h
d9fcdc742e0118ccfc12d5f5f5f3c043  ./include/messenger.h
ac6862447751e9b7b12c59c6aa491011  ./include/nameserver.h
6012cd83a92ded693f863eb5dff23be8  ./include/ourio.h
66c074d7f5a93750ed3cd4e1796ebc33  ./include/ourlib.h
ad8404d6ff9b3cd23ab35da21a70037f  ./include/priorities.h
e1b272be0185c041b8dcf2d5df968f7a  ./include/project.h
ad16b7ff18670d6a6a5afd2236d41650  ./include/queue.h
f42681e19e3dc50960b701be52a53546  ./include/rps_server.h
0e390e153530b05118c87063a5dd3120  ./include/run_tests.h
621140a6884a581a7e896c056efe717e  ./include/scheduler.h
d0a3e405c2b9edacbe2561ccd872b603  ./include/sensor.h
04f3437f5e8c454f6be806b30a0f186a  ./include/sensor_server.h
2a521c524bcdc6a4e301e9a923dc136d  ./include/simple_sm.h
7425e47fb975228ae9a05887462d1b23  ./include/string.h
7a1b1692e08413e0632172e8bab8cd11  ./include/strings.h
fdb9f38ec11ac3afd4644d93d582c008  ./include/syscall.h
a1acf7c44412bd034c8932b9df952c9b  ./include/task.h
cd42684f05820c5322c424f711652b5f  ./include/test_helpers.h
077b0816d3b8542fd64497f05d360c41  ./include/timer.h
95a649f7921abd9d08642eb541c5c995  ./include/reservation_server.h
9830e1eafae7ba84b37489351892c1a4  ./include/track_node.h
```

```
4a954411751380de42180fe5cd60c7f6  ./include/track_data.h
473d468b5c307bb6eeaeca7d35acec54  ./include/ts7200.h
04e5789e07fb586a1b6b101acf73a96b  ./include/uart.h
d12fe2880a682aa19c08f51ac05ceb26  ./include/user_prompt.h
f703df970dc0bc8279d9e4ae5b5298f4  ./include/calibration.h
7ffedb06b94e0ba0050c3a42269be680  ./include/distance_server.h
2b1154e2e313f9bc358a3a22cceb5f5d  ./include/location.h
eefb867d28328c48d1f87e66b176ebbf  ./include/physics.h
860b3c773db3e6c1a05693985ea72eb1  ./include/timings.h
67795f2f9468209a9c5758228e280f1e  ./include/track_edge_array.h
8e2f5f00ae304ef9738376b157ee5ee1  ./include/train.h
d03b8d04bdad4e70525f2b961203302a  ./include/switch_server.h
dd2b4cd6cd2e5b4275b864749bdfdb40  ./include/demo.h
bde87554c6753ab69679a0096a9f0eab  ./kernel/interrupt_handler.c
0a86f08289c856a47d8ca56ee92a7fdb  ./kernel/kernel.c
ab372eec3ca8c8429171c6ff462c23fa  ./kernel/main.c
e9e1ac4c766842ccd536393e5915b901  ./kernel/messenger.c
8108e4058969ac8a377ad7241006656c  ./kernel/scheduler.c
21dcdc6ab513c96b214e074b17836870  ./kernel/task.c
aed6ada1a6bce02e32269bb41ab1bf58  ./ourio.c
06c596f61b859cd9a572323c4d1b0ae1  ./uart.c
6006d09d2d94833c991897dda725e5df  ./syscall.c
447725a2507da0fefd11d3bfcfdb65bb  ./physics.c
db794839ecaf9c135161e36d45ed24b7  ./timer.c
03405297ee2aa55f6320405e437838b8  ./project/calibration.c
940c64a85e0ce0be2e92cdec7a6c3186  ./colorgcc
ad4bab3f9b64a0a6297df47541d1d698  ./clockserver.c
114ddacba047bcaa3cfef9925015a4e5  ./scripts/spread.c
0f82872b40cd19b7541c0d8ebe6c8602  ./idle_task.c
82a620d148aec576d56b0dd3b97b8a1a  ./location.c
65c110e4d7da151a5d2c0763f8e338d3  ./timings.c
6d621ccc191ee6a634d64b880f69edab  ./switch_server.c
988f42698f988d7ea7cdb30d26cba724  ./demo.c
df844325885708ff93e4eafdd2f12a51  ./unittests/all_tests.c
ad417e31447983fe13bab3ad5f7d4fac  ./unittests/bitmask_tests.c
9e27f3bff6356af0d99ee24d6c7c2db9  ./unittests/linked_array_tests.c
1413c01a1b7694350c45058180c18a5b  ./unittests/heap_tests.c
f068e72416d05f20660390bb1e50a4ef  ./unittests/test_helpers.c
24bc70d54528b73d8d7eb23f573921bc  ./unittests/test_helpers.h
4b935cf081a780b986545df31445c30a  ./unittests/strings_tests.c
c178f53bceae14123574a76460d4a1d9  ./unittests/dijkstra_tests.c
b625a0bbf0fa9e81c70dd5648f206b3a  ./unittests/free_tests.c
eb4742a2a697b95c442ccedfb5fee408  ./sensor.c
d141a877fdf638c446d299532e5db186  ./priorities.c
0077ca5ad3b96616b9d58756ba4bb89a  ./string.c
acedd21cd25515e2b43bc6ecaed28c72  ./tests/scheduler_speed_test.c
```

```
0a009e1c2cf45b3a2e4fe54aa323bf7a  ./tests/task_creation_errors_test.c
fb8849e3daee133a6dbd8d46aae51841  ./tests/multiple_priorities_test.c
eee31a1660aeb081646807700f42916e  ./tests/assignment_1_test.c
bcb10795dece42854f1afdfeba58abc3  ./tests/message_passing_test.c
36f8e02155a2ec078a6b9f39c7fa84c6  ./tests/basic_test.c
e84e447e74452e7a0e121584432b0101  ./tests/nameserver_test.c
70ba721afbb029c5f5fd274610cd04e5  ./tests/hwi_test.c
ba53b95ab3af146a4e579451899bba5b  ./tests/srr_speed_test.c
9baef3b3aba60fa62a52f1fac6081ee5  ./tests/rps_server_test.c
6525c92955cc2dcefd59010920d3def1  ./tests/syscall_speed_test.c
3efd35b657aae300b4c27610c3de0ed1  ./tests/assignment_3_test.c
55d83782fdbdbd446d40150475e99a64  ./tests/clockserver_test.c
c83a8c9eb4d8bfd8ce93024e85e90615  ./tests/uart1_intr_test.c
9405cd8cffd7436c9d32ca14e651ab03  ./tests/train_test.c
```

3 Project Description

3.1 Reservation Server

The reservation server is responsible for handing out and managing reservations of track edges. Each edge has an associated "edge group" that encapsulates the idea that some edges must be reserved at the same time. The following edges are in the same edge group:

- A edge and its reverse edge
- Two edges on a branch node
- Edges from BR153 and BR154 or edges from BR155 and BR156

When a train requests to free/reserve an edge, the reservation server frees/reserves all edges in the same edge group.

When a train attempts to reserve an edge that is reserved, the reservation server remembers that train and notifies it when the edge becomes free. This allows us to prevent having to poll the reservation server.

3.1.1 Track Edge Arrays

To manage reservations of edges, we use a track edge array to map edges to an index into a large array. This is currently implemented using an array of size `TRACK_MAX * 2` since each node has at most 2 edges. This is very space inefficient as there are many more nodes with 1 edge than there are with 2 edges. However, this was the simplest solution and we're not space constrained.

3.2 Deadlock Detection/Resolution

A deadlock occurs when either both trains are blocked on an edge or if one train is blocked on an edge and the other train isn't moving. This is constantly monitored in the train controller whenever it receives a location update.

When a deadlock is detected one of the trains is rerouted so that it doesn't use the edge that it is blocked on. This works for now because we only have 2 trains and we avoid the bays.

3.3 Path Following

One main change has been made to path following. Trains now reserve edges up to 70cm ahead of them and free edges past 15cm behind them. Reserving is done by just looking 70cm ahead in the path it has, and freeing is done by doing a DFS backwards from its curent location. The DFS was chosen because it is safer and simpler than trying to keep a queue of reserved edges and popping from it. Each train tracks which edges it has reserved, so that it doesn't send unnecessary requests to the reservation server.

3.4 Stopping Distance

In TC1 it felt good conceptually to do stopping like we did accelerating. That is, we had a deceleration function, and after so and so many ticks, we say we've stopped when current velocity was 0. If we wanted to change the stopping distance, manipulating the deceleration function to produce that stopping distance was notoriously difficult. We dropped the deceleration function and instead focussed entirely on just updating the train by the calibrated stopping distance. As a result, the location server is error-prone while stopping, but when fully stopped it is incredibly accurate. As a result, our path following actions always require a full-stop instead of slowing down.

3.5 Dijkstra with Blocked or Broken Edges

Dijkstra is performed as usual. Before traversing an edge, we can do $O(1)$ lookups into our broken or blocked tracked edge arrays and refuse to traverse those edges.