Train Control 2

Ronuk Raval (20345750) & Christophe Biocca (20322763)

July 17, 2012

1 Files and compiling

The source code is available under `/u/cbiocca/cs452/tc2/src/`, while the executable is located at `/u/cbiocca/cs452/tc2/kernel.elf`. It can also be built by running `make prod` in the `src` folder. Here are the file checksums.

```
fed337393a008e3a8b936e9de99bd26a  ./kernel.elf
83185b7100ee5f0d25ac34c0d3d584d3  ./src/.gitignore
4685c312e1797a76e744e98e8c74c353  ./src/Makefile
436afe7fc14408ee2f8be159b83c4ac1  ./src/README.md
0d7272fd2615423d956ebf7b375929d8  ./src/data/BEST_HASH
7b6addd188d4401150260c009882f7dc  ./src/data/parse_track.py
ba66a1256e252e3ee3f38ad235e61b15  ./src/data/parseoutput.rb
2f6395e2f4c680f79e35804b969c4f54  ./src/data/tracka
87f6c266346146d0763deef88dac23c6  ./src/data/trackb
943217198c8f49c0ba9fea5d40f6baf8  ./src/doc/CALIBRATION
d761266f3eb599d0dfc6dd88d52619c9  ./src/doc/WORKFLOW
5fb080eae0d5af66c80b275837d829c6  ./src/doc/k1.lyx
af526ee1c1869d021c620b322b9f7e25  ./src/doc/k2.lyx
691be4bc843ac1b54e2e701a98b4fd81  ./src/doc/k4.lyx
acd7d9afde797dda0c3d097569d301af  ./src/doc/tc1-hack.lyx
1b5890fbf8d8274f9e3dcdd2ebfb12ea  ./src/doc/tc1.lyx
3fb369afd0a490365655a5bfa157adf8  ./src/include/bwio.h
c13942a5833307674dba2f254809595d  ./src/include/cpsr.h
fc6c1c205ff7a1c7fbb1b17de16b278a  ./src/include/debug.h
91ee8f0b6de1f3e23857502e3732c852  ./src/include/kernel/interrupts.h
f48ea9097d647cf0e299eea0cb838117  ./src/include/kernel/ipc.h
04095eebff3a2559d09e16cd4e998617  ./src/include/kernel/task.h
2abdf733846988a172af7f34e2051152  ./src/include/kernel/trampoline.h
aeb47082076d178c1784cc5dcd786a2b  ./src/include/lib.h
7ccf926195225fcab016186c8c5185fe  ./src/include/stdbool.h
ab50e9674e5eb7aa23b97d35028320fa  ./src/include/ts7200.h
1a026162a95f84eb28bfae5e8c50c429  ./src/include/user/clock.h
b11121fa5217f977a0441d8721f0520d  ./src/include/user/clock_drawer.h
```

```
f6fb657b597079f5a7821c54e8b24962  ./src/include/user/controller.h
dbdf4d53d41168a98bd51fd9ea4997f3  ./src/include/user/courier.h
a01c57a358186624a20d6b392e14243b  ./src/include/user/engineer.h
e9a7384b7de24bf6eca953fb6555e0c0  ./src/include/user/heap.h
c43f99b170424b893e6769414cca0292  ./src/include/user/init.h
e79cdfc967878c672f7ce6febe58c64f  ./src/include/user/kinematics.h
caf77700473316bef55fe272ecbcfc32  ./src/include/user/log.h
af578ce26dc1a2dd92828b10bfe1175e  ./src/include/user/mio.h
1f083cc3eb8d2b6cc7fe0467b7bd871b  ./src/include/user/nameserver.h
1133e3d3b68d8ddfc68f48b6de2a0112  ./src/include/user/parser.h
7df7791d51a4d0ee526cce4573eccadc  ./src/include/user/pathfinding.h
3e594dfbd6e38adb10db052421b7dbb7  ./src/include/user/priorities.h
65dc14ad865906588c2f844ef3761116  ./src/include/user/sensor.h
ee296db2934aa57550a07af53aedcb24  ./src/include/user/string.h
403814223eb206dec10dd4796c930b42  ./src/include/user/syscall.h
6948a47691ec5fe21fe91d8026a00fa3  ./src/include/user/tio.h
936d372675c83ab6ace0c553594580cf  ./src/include/user/turnout.h
08ae8d519a47df09d891402ef4480e64  ./src/include/user/vt100.h
25757ddee6d82a16be926075e04b4972  ./src/kernel/bwio.c
3880bd8c7594d3b0b5f11e55a7c0f41d  ./src/kernel/interrupts.c
3ebd6f0e170452eda4094e49b2bc7507  ./src/kernel/ipc.c
ef8aa6a3a2ebdfa30f4f52fb9c9306df  ./src/kernel/kernel.c
b6927156163cab78bc599a7cc1e76949  ./src/kernel/lib.c
a5551cb2004e046f38ebd83a618db810  ./src/kernel/task.c
ee3f874c894c039f95edbbf2e1350e05  ./src/kernel/task_internal.h
ba2bbda6474d8eb0c742de1778c353d1  ./src/kernel/trampoline.c
85a06c179e618d41cfa86f5a51346f43  ./src/linker.ld
ee9847d34ef3f14582e58f2bbdb3c2da  ./src/user/clock.c
ae9d089d2f5260d6f2ecaea987e19b67  ./src/user/clock_drawer.c
f86432368262c6f69ceb08d569fa89e2  ./src/user/controller.c
5db228d5d3288c110d866f049d6f6f94  ./src/user/courier.c
59b50d4f3ac767fe936bf24847a16df3  ./src/user/engineer.c
49d525d039bf4822d54c04c534c79901  ./src/user/init.c
5daec5245db7b73e524a6a4ed8f228a3  ./src/user/kinematics.c
63e1057f133f97319ac52a7724e60737  ./src/user/log.c
10d8149abda2c2a5247caadb4118484d  ./src/user/mio.c
1056255ba9ee68987414d82da357f47e  ./src/user/nameserver.c
38d9194e75203bc4427f18ef147ebc95  ./src/user/parser.c
21c33d262d8d67ca9bc278c21554a7f0  ./src/user/pathfinding.c
bedac67e6bda1fd538f0ae710a9e1bdf  ./src/user/sensor.c
33da1c0024215d6c4469d6050e90b91a  ./src/user/syscall.c
dabfe5cebd5955c1f839def8cf457d4b  ./src/user/tio.c
68ddaf08a986158a67985af6b6729086  ./src/user/turnout.c
eea47287ff66b4bc33eb8d2da57c93ca  ./src/user/vt100.c
```

2 Reservations

Reservations are implemented as 2 independent halves by the train engineer task: respecting reservations on the track and making and releasing reservations as the engineer needs them. In the reservation system implemented, the granularity supported is limited to track edges only. For the purposes of the reservation system, a track edge and its reverse are treated equivalently.

2.1 Respecting Reservations

All the track edges are available as part of the global track graph, and is thus readable by all engineers. An extra field called `reserved` was added to the track edge data type, which stores the train ID that the edge is allocated to or `-1` otherwise. Thus, the invariant that all engineers must maintain is simple: one must never traverse an edge that they do not own. Since the state of reservations is available globally, checking reservations is cheap and doesn't even require a `Send()`/`Receive()` cycle.

When an engineer approaches a piece of track they do not own, they slow down and check the status later. An engineer must always ensure that they are capable of stopping without violating reservation invariants. If an engineer comes to a complete stop, they set a timeout which has both a static component (8 seconds) plus an additional random component based on the train ID (train 0 adds 0 seconds, train 80 would add 22.5 seconds). If this timeout is hit and the train is still unable to get the reservations it needs to follow the planned path, it replans its route. Route planning takes into account the current state of reservations and plans around it.

The random component based on train ID allows for fairly intelligent emergent behaviour from the trains. Since there is a large variance between the timeouts, in a deadlock situation, it is very likely that a single train would choose to replan before the other train. This first train would then follow its alternate route, giving up reservations which can then be used by the other train.

2.2 Making and Releasing Reservations

All updates to the reservation information in the track graph are made by the train controller, which is a single task. This synchronizes access to the graph and avoids an entire class of concurrency issues.

The engineer keeps track of reservations in three different classes. First, a function computes the set of edges that it thinks it needs. It does so by adding the edge that it is currently on, as well as all the edges its tail might be on if its near the ends of the edge. Finally, it projects forward by the current stopping distance, adding all traversed edges to the set, as well as edges on both sides of traversed branches.

Then a function compares what is in the set that is needed and the information presented in the track graph. If it detects that certain edges that it needs are are assigned to it, it moves said edges to a new set of granted reservations.

Finally, a function computes edges that are in the granted set but are no longer in the needed set. These edges are released back to the controller.

All the while, a courier is used as a communication bridge between the train controller and the engineer.

2.3 Interaction with Track Features

The train makes primary and secondary expectations (sensors that when triggered should be passed along to this specific engineer) with the train controller. Two engineers expecting a sensor is strictly an error and manifests itself as an assertion failure. The idea behind setting primary and secondary expectations is allow for track failures, should a sensor fail several times the controller can detect it and modify the track topology accordingly. For malfunctioning switches, the notion of an alternate sensor is also supported, and the switch would be marked as disfunctional should an alternate sensor be triggered enough times. However, at the time of writing, these systems are only partially implemented and no modifications to track topology are actually made.

To synchronize control of various track features, an engineer is only allowed to set an expectation on a sensor if it owns reservations on the edges on both sides of said sensor. Likewise, an engineer may only switch a branch turnout if it owns all 3 edges. This conservative system prevents quite a few accidents since expecting a sensor expressed the notion that a train rightfully intends to pass over it and thus can only do so if it has the reservations to back it up. Similarly, a turnout should not be switched unless the train is immediately able to pass over it.