Όνομα: Ανδρέας Θεολόγος Σπανόπουλος
ΑΜ: 1115201700146

Λίγα λόγια για την εργασία

- Έχουν υλοποιηθεί όλα τα ζητούμενα της εκφώνησης.
- Η δομή της κοινής μνήμης βρίσκεται στο αρχείο shared_memory.h, και στο shared_memory.c αρχικοποιούνται οι μεταβλητές και οι σημαφόροι.
- Στο σχήμα μπορείτε να δείτε μια αναπαράσταση του σταθμού και των μεταβλητών του.
- Έχω φτιάξει το πρόγραμμα mystation, το οποίο είναι υπεύθυνο για τη δημιουργία όλων των άλλων διεργασιών.
- Έχω προσθέσει μια έξτρα παράμετρο για τη δημιουργία ενός λεωφορείου: την "-i id", που ουσιαστικά είναι η ταυτότητα του λεωφορείου.
- Οι διεργασίες station-manager και comptroller τερματίζονται μόλις αποχωρήσουν όλα τα λεωφορεία από το σταθμό. Το πόσα λεωφορεία θα δημιουργηθούν υπάρχει στο configfile.
- Οι κοινή μνήμη (shared memory) αποδεσμεύεται από τη διεργασία mystation, μετά τον τερματισμό όλων των υπολοίπων διεργασιών.
- Γενικά υπάρχει αρκετός hard-coded κώδικας. Ειδικά οι συναρτήσεις που διαβάζουν και ελέγχουν τις παραμέτρους γραμμής εντολών (get_input()).

Θα προσπαθήσω να κάνω σύντομο αυτό το README επειδή

1. Δεν θέλω κουράσω τον αναγνώστη.
2. Έχω να κάνω εργασία ΥΣΒΔ που λήγει αύριο και εργασία τεχνητής που λήγει μεθαύριο.

Ουσιαστικά, το “ζουμί” του προγράμματος βρίσκεται στο συγχρονισμό των λεωφορείων και του station manager (mystation και compotroller τόσο εύκολα που δεν τα αναφέρω καν).

Πως λειτουργεί ο station manager?

Αφού διαβάσει το input του με χρήση της συνάρτησής του get_input(), μπαίνει σε ένα while loop. Από το loop αυτό θα βγει όταν αποχωρήσουν όλα τα λεωφορεία που δημιουργεί η διεργασία mystation. Με το που μπει στο βρόχο, μπλοκάρεται σε ένα σημαφόρο (**station_nanager_mutex**) ο οποίος αρχικοποιείται με 0.

Ο μόνος τρόπος για να ξεμπλοκαριστεί, είναι να τον "ξυπνήσει" κάποιο λεωφορείο που είτε θέλει να μπει στο σταθμό, είτε να βγει από στο σταθμό, είτε να τον ενημερώσει για το πέρας κάποιας μανούβρας του. Προφανώς, στην αρχή του προγράμματος κανένα λεωφορείο δεν έχει προλάβει να εισέλθει στο σταθμό, οπότε μόνο ένα εισερχόμενο λεωφορείο μπορεί να τον ξυπνήσει.

Υπάρχουν 4 λόγοι για να ξυπνήσει ένα λεωφορείο τον station manager:

1. Ζητάει άδεια εισόδου στο σταθμό και αναμένει να του υποδείξει ο station manager που να παρκάρει.
2. Θέλει να ενημερώσει τον station manager πως ολοκλήρωσε τη μανούβρα παρκαρίσματος.
3. Ζητάει άδεια εξόδου από το σταθμό και περιμένει να του τη δώσει ο station manager.
4. Θέλει να ενημερώσει τον station manager πως ολοκλήρωσε τη μανούβρα αποχώρησης.

Για να μπορεί να διακρίνει ποια περίπτωση έχουμε κάθε φορά που τον "ξυπνάμε", χρησιμοποιούμε flags που έχουν οριστεί στο shared segment.

- Στη πρώτη περίπτωση, ψάχνει, να βρει αν υπάρχει διαθέσιμη θέση στάθμευσης για το συγκεκριμένο λεωφορείο. Αν υπάρχει τότε το ξεμπλοκάρει (αυτό έχει μπλοκαριστεί σε έναν σημαφόρο, τον **park_mutex,** που θα αναλυθεί αργότερα), και του "λέει" που να παρκάρει.
- Στη δεύτερη περίπτωση, αφού το λεωφορείο έχει παρκάρει, το οποίο συνεπάγεται πως έχει ολοκληρώσει τη μανούβρα του, ξεμπλοκάρει το επόμενο λεωφορείο από τον σημαφόρο **entry_queue (**και αυτός θα αναλυθεί αργότερα), και επικοινωνεί μαζί του για να δει αν υπάρχει διαθέσιμη νησίδα για σταύθμευση.
- Στη τρίτη περίπτωση, ελέγχει αν υπάρχει κίνηση εξόδου στο σταθμό, και αν δεν υπάρχει τότε αφήνει το λεωφορείο να εξέλθει (δηλαδή το ξεμπλοκάρει από το σημαφόρο **exit_queue** που θα αναλυθεί σε λίγο).

- Στη 4η περίπτωση, ενημερώνεται από το εξερχόμενο λεωφορείο πως αυτό ολοκλήρωσε τη μανούβρα του, και με τη σειρά δίνει άδεια στο επόμενο λεωφορείο που αναμένει για άδεια αποχώρησης, να φύγει.

Να σημειωθεί πως σε κάθε περίπτωση αλλάζουν οι τιμές σε διάφορα flags για να μπορεί να κατασταθεί δυνατή η επικοινωνία station manager και λεωφορείων.

Πως λειτουργεί η διεργασία bus?

Ακριβώς όπως ορίζεται στην εκφώνηση της εργασίας:

1. Άφιξη και αναμονή έξω από σταθμό για εξυπηρέτηση από τον station manager.
2. Ενημέρωση από station manager για ασφαλή είσοδο στο σταθμό και σε συγκεκριμένη νησίδα.
3. Άφιξη στην νησίδα που έχει οριστεί και αποβίβαση.
4. Ενημέρωση ledger για τον αριθμό των αφίξεων.
5. Παραμονή στο χώρο στάθμευσης και επιβίβαση επιβατών.
6. Αίτηση για αναχώρηση από σταθμό και αναμονή για την εξυπηρέτηση από τον station manager.
7. Ενημέρωση από station manager για ασφαλή αναχώρηση.
8. Αναχώρηση και είσοδο στο δημόσιο δρόμο.

Επεξήγηση συγχρονισμού και επικοινωνίας διεργασιών

1) Σχετικά με την είσοδο λεωφορείων στο σταθμό.

Για την επίτευξη της ομαλής και ελεγχόμενος εισόδου στο σταθμό από συγκεκριμένο αριθμό λεωφορείων, χρησιμοποιούμε 2 σημαφόρους:

- sem:entry_queue <-- 1
- sem:park mutex <-- 0

Ο πρώτος λειτουργεί σαν “ουρά” για τα λεωφορεία. Αφού έχει αρχικοποιηθεί με 1, μόνο το πρώτο λεωφορείο θα περάσει στην αρχή εκτέλεσης του προγράμματος. Τα υπόλοιπα θα μπλοκαριστούν σε αυτόν. Όταν ένα λεωφορείο ολοκληρώσει τη μανούβρα παρκαρίσματος, ενημερώνει τον station manager, και αυτός με τη σειρά του εφαρμίζει V() στο σημαφόρο για να ξεμπλοκαριστεί το επόμενο λεωφορείο.

Όταν ένα λεωφορείο “περάσει” από τον πρώτο σημαφόρο (δηλαδή προχωρήσει μετά τη πράξη P() σε αυτόν), προχωράει στον δεύτερο.
Λόγω της τιμής αρχικοποίησης του πρώτου σημαφόρου entry_queue, μόνο ένα λεωφορείο μπορεί να είναι μπλοκαρισμένο στον δεύτερο σημαφόρο park_mutex. Όταν ένα λεωφορείο μπλοκάρεται σε αυτόν τον σημαφόρο, θεωρούμε ότι βρίσκεται σε επικοινωνία με τον υπεύθυνο του σταθμού, και του ζητάει άδεια για να εισέλθει στο σταθμό. Ο υπεύθυνος σταθμού θα του επιδείξει τη νησίδα στην οποία θα πρέπει να πάει να σταθμεύσει μόλις υπάρξει διαθέσιμη θέση.

Ο λόγος που χρησιμοποιείται ο δεύτερος σημαφόρος είναι για να ”απομονώνουμε” τα λεωφορεία, με σκοπό να εξυπηρετούμε ένα κάθε φορά. Αποθηκεύουμε τον τύπο του σε μια προσωρινή μεταβλητή στη κοινή μνήμη, και όσο αυτό είναι μπλοκαρισμένο, ψάχνουμε διαθέσιμη νησίδα που να μπορεί να το δεχτεί.

Με άλλα λόγια ο πρώτος σημαφόρος ελέγχει τη κίνηση εντός του σταθμού, και ο δεύτερος σταματάει το λεωφορείο μέχρι να του δοθεί άδεια να συνεχίσει.

2) Σχετικά με την έξοδο λεωφορείων από το σταθμό.

Για την επίτευξη της ομαλής και ελεγχόμενης αποχώρησης λεωφορείων από το σταθμό, χρησιμοποιείται ένας σημαφόρος:

- sem:exit_queue <- - 0

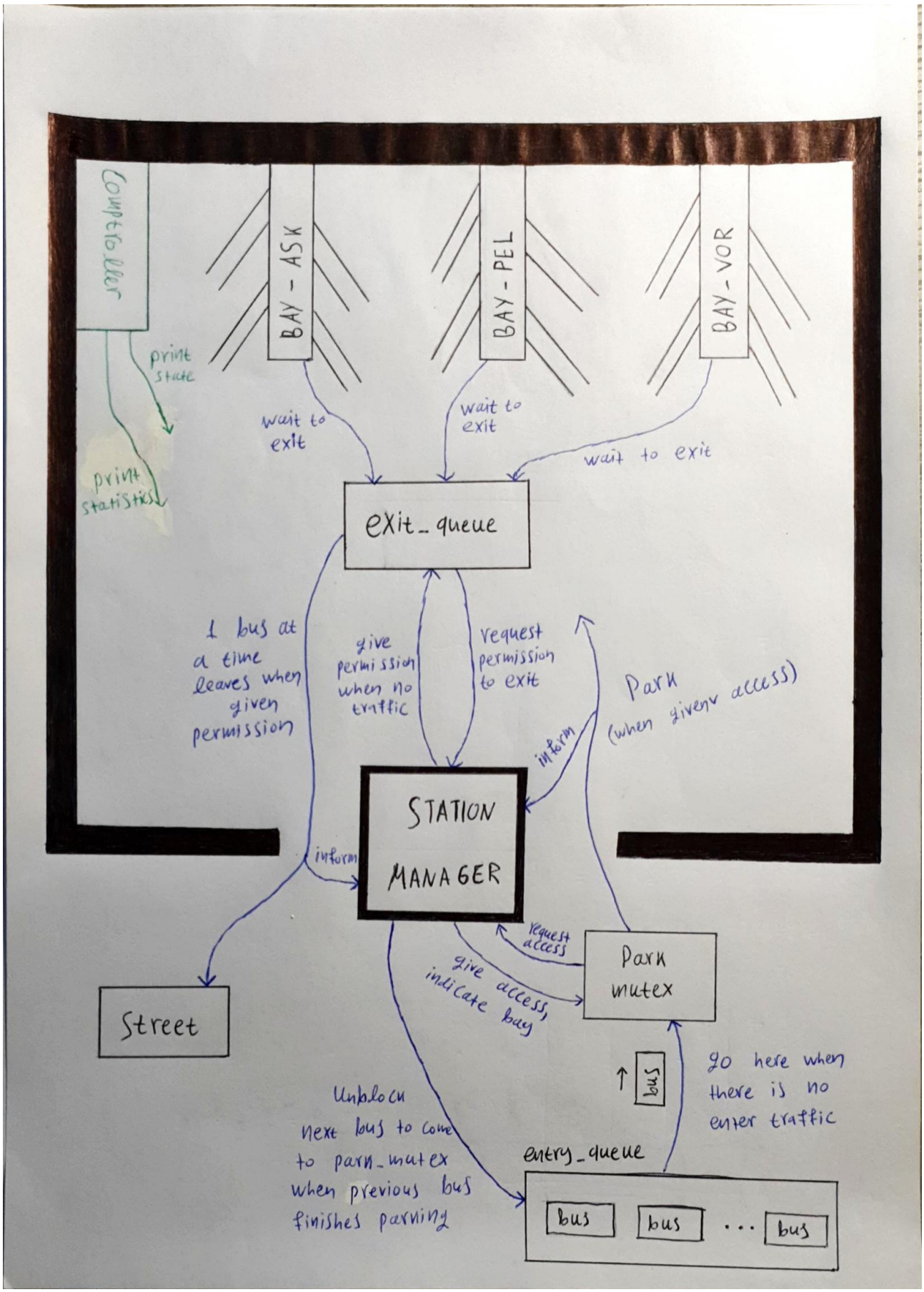
Όπως και με το σημαφόρο εισόδου entry_queue, έτσι και αυτός ο σημαφόρος λειτουργεί σαν ουρά για τις διεργασίες λεωφορείων.
Τα λεωφορεία μπλοκάρονται στον σημαφόρο αυτόν, και ο station manager τα ξεμπλοκάρει όταν βεβαιωθεί πως δεν υπάρχει κίνηση εξόδου επί του σταθμού.

Στην περίπτωση της εξόδου δεν χρειαζόμαστε δεύτερο σημαφόρο επειδή δεν έχουμε ανάγκη να απομονώνουμε να λεωφορεία για να δούμε τον τύπο τους. Δεν υπάρχει κάποια προτεραιότητα όσο αφορά τη σειρά εξόδου λεωφορείων.

3) Άλλοι σημαφόροι και λεπτομέρειες.

Χρησιμοποιώ κι άλλους σημαφόρους για να διατηρήσω την ακεραιότητα του Critical Section και για να μην γίνουν ταυτόχρονα writes ή writes/reads στην ίδια μεταβλητή κοινής μνήμης. Θα τους ανέλυα όλους αλλά δεν έχει νόημα, καθώς έχω προσθέσει υπέρ-αρκετά σχόλια στον κώδικα.

Παρακάτω παραθέτω μια σχηματική αναπαράσταση του σταθμού και του τρόπου επικοινωνίας και συγχρονισμού των διεργασιών.


Controller
print state
print statistics
BAY - ASK
BAY - PEL
BAY - VOR
wait to exit
wait to exit
wait to exit
exit_queue
1 bus at a time leaves when given permission
give permission when no traffic
request permission to exit
Park (when given access)
inform
STATION MANAGER
inform
Street
request access
give access, indicate bay
Park mutex
bus
go here when there is no enter traffic
Unblock next bus to come to park_mutex when previous bus finishes parking
entry_queue
bus
bus
...
bus

Άλλες χρήσιμες πληροφορίες

- Ο κώδικας έχει αναπτυχθεί στον κειμενογράφο Atom. Κατά συνέπεια, οι διάφορες ευθυγραμμίσεις σχολίων και κώδικα έχουν γίνει βάσει των default ρυθμίσεων του Atom. Συνεπώς, θα πρότεινα να χρησιμοποιηθεί αυτός ο κειμενογράφος για την ανάγνωση του κώδικα.
- Για την μεταγλώττιση των προγραμμάτων, απλά δώστε την εντολή make στο terminal.
- Για να ξεκινήσετε το πρόγραμμα, απλά δώστε την εντολή ./mystation -l Configuration_File.txt.
- Για να διαγράψετε τα αντικείμενα και τα εκτελέσιμα αρχεία, απλά δώστε την εντολή make clean στο terminal.
- Για να διαγράψετε το Log αρχείο που δημιουργείτε για να καταγραφούν οι διάφορες δραστηριότητες που λαμβάνουν χώρο εντός του σταθμού, απλά δώστε την εντολή make rm_log στο terminal.
- Σχετικά με το Configuration File, όλες οι παράμετροι μπορούν να αλλαχθούν για να αλλάξει η “είσοδος” του προγράμματος, εκτός από τον αριθμό των bays (που είναι 3 by default), και τον αριθμό των spots, ο οποίος έχει οριστεί να είναι ίσος με 4 στο shared_memory.h με χρήση της μακροεντολής #define. Αν θέλετε να αλλάξετε τον αριθμό των θέσεων (spots) ανά νησίδα, απλά αλλάξτε τη τιμή του #define στο αρχείο shared_memory.h.