

SWEN-563 EEEE-643 CMPE-663

Real time embedded systems

Project 2b

Dinesh anand bashkaran – dab8730@g.rit.edu

Zachary Weeden – -zdw7287@rit.edu

Content:

AREA OF FOCUS

ANALYSIS

PLAN

PROJECT RESULT

LEARNING

1. AREA OF FOCUS:

We did split the work equally. We have good understanding as teammates. Coding, debugging, program flow was discussed and we contributed to implement it.

2. ANALYSIS:

Introduction:

The project is all about utilizing the multi-tasking feature of QNX. This is done by controlling two servo motors in application based. The servo motor functionality is similar to 2a. The motor could be moved to 6 positions with respect to duty cycle defined. Pulse of 20ms is used to control the motor.

When it comes to the control over motor, the user could control it anytime. Just in case if system doesn't get any user input, the pre-loaded recipe runs.

Recipe/program flow:

The PWM signal is configured using the neutrino box. PWM input is configured in 2 channels of neutrino box to make the motor operate with respect to the input. The Difference between the total time and the elapsed time is taken to calculate the sleep time.

STATE MACHINE: state machine runs through different positions of the servo motor. Each of them is pre-defined with different duty cycle in the 20ms period time. Initially, the program starts from "starting" where the program checks for user input. This state is "check input" where the program keeps track on the UART if any input is received. If the user input is defined, the respective action is performed. If the program finds nothing from UART, it just runs the recipe which is "RUN STEP". The recipe depends on current recipe state, just in case if the recipe finds a user input, the input is parsed. But generally, the program

is implemented to pause it anytime during run period and perform user defined actions.

OPCODE: The opcode is of 8 bit unsigned integer. 3 bit denotes the action and 5 bit the value. To parse the action, the opcode is shifted right by 5 times. The captured opcode is check with respective actions which is defined in if statements.

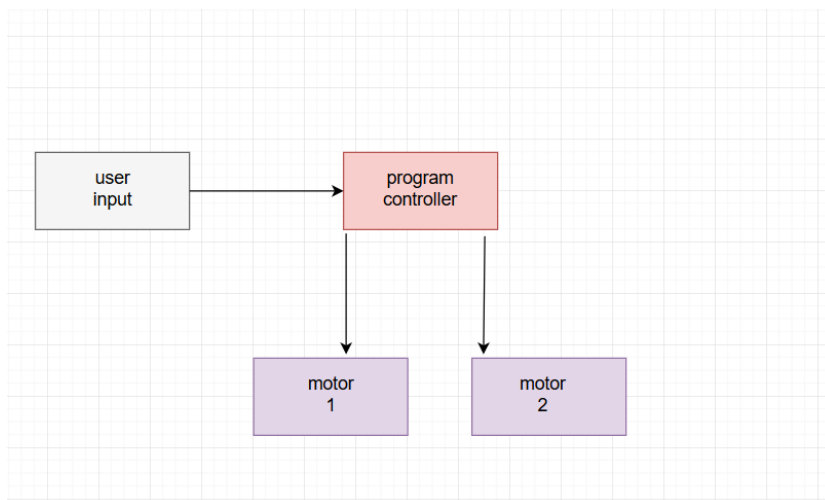
Test plan:

The project was similar to 2a except the part where certain things have to be configured in software. We were confident about the running of recipe. All we had to worry was to generate the pwm from QNX software. The basically, we verified the recipe function with the servo motor. To check the wrong functionality, wrong opcode was given. This wrong opcode is not defined in if statement, so the program was not able to detect it and ended in error at that statement.

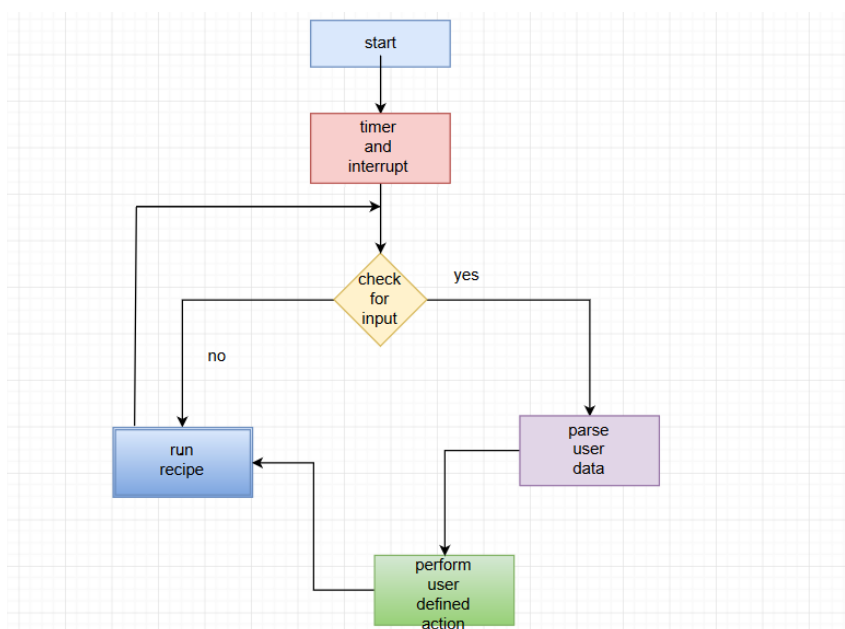
Comparison and learning:

I would say the project 2b was more like software programming. 2a was more like hardware. I got a chance to learn certain things and it enabled me to differentiate how same things could be done in software and hardware. In 2a, pretty much from timer configuration, generation pwm signal had to deal with registers and all hardware connection such as GPIO, preload enable and flags etc. STM32 was used to do all these monkey business.

However, 2b was quite different. Everything was done in software. It also took less time because we didn't have to check the hardware manual again. From my view, I would say software is easy to implement. The hardest part is where we obtain the pwm signal from QNX neutrino box. We pretty much had other things up and running from 2a so it was not that hard.

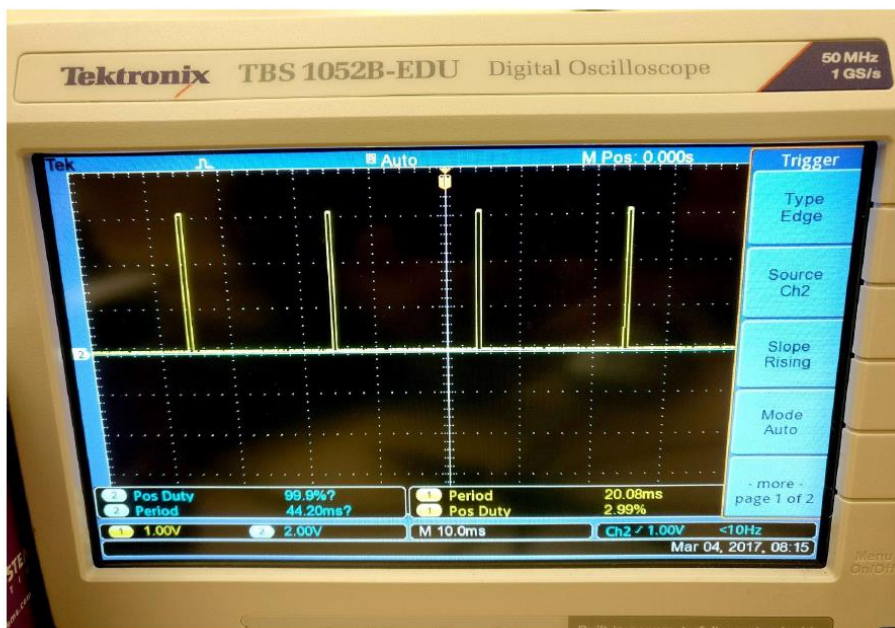
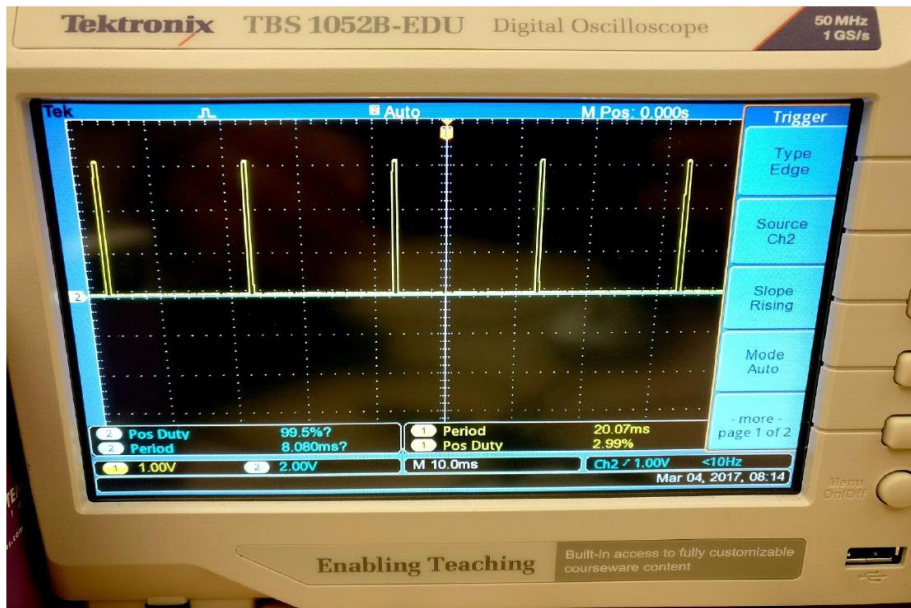


Control Flow

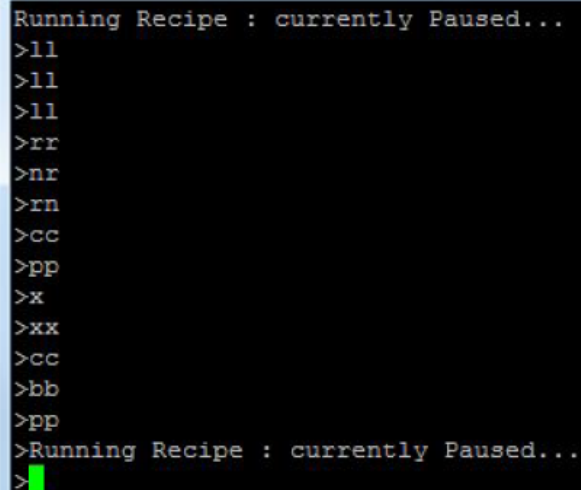


Program Flow

Results:



PWM generated from QNX neutrino box a and b. The pwm is generated with duty cycle 20ms.



```
Running Recipe : currently Paused...
>ll
>ll
>ll
>rr
>nr
>rn
>cc
>pp
>x
>xx
>cc
>bb
>pp
>Running Recipe : currently Paused...
>
```

This screen shot of the output terminal shows the results and the performance of the servo motors with respect to the user defined commands as well as the recipe. Like we discussed, the program is initially waiting for the user input to either run the recipe or specific function given by the user. The program reads it through UART and does the job.