

Microcontrollers Research Final

**Program:**

***Course Code: EPM282***

***Course Name: Power Electronics and Drives***

***Examination Committee***

**Dr. Sherief Hammed**

**Ain Shams University**

**Faculty of Engineering**

**International Credit Hours Engineering Programs (I-CHEP)**

**Spring Semester – 2020**

**Student Personal Information for Group Work**

|  |
| --- |
| **Student Names:** |
| George Gerges |
| Fady Mina |
| Omar Zohdy |
| Abanoub Tarek |

|  |
| --- |
| **Student Codes:** |
| 17P8181 |
| 17P8187 |
| 17P8190 |
| 17P8196 |

**Plagiarism Statement**

I certify that this assignment / report is my own work, based on my personal study and/or research and that I have acknowledged all material and sources used in its preparation, whether they are books, articles, reports, lecture notes, and any other kind of document, electronic or personal communication. I also certify that this assignment / report has not been previously been submitted for assessment for another course. I certify that I have not copied in part or whole or otherwise plagiarized the work of other students and / or persons.

**Signature/Student Name:**

**Fill in**

**Date:**

**Fill in**

**Submission Contents**

**01: Functions**

**02: Finite State machine**

**03: Non-merit functions**

Functions:

void PAUSE ():

It basically pauses the timer and maintain the number of seconds so we can resume it anytime.

uint32\_t ATI (char I [10]):

It changes an array of char(strings) to an integer.

void STOP ():

It clears the timer and disable the timer.

ENDSEQUENCE ():

it flashes the red led 5 times and returns to idle state.

MICROWAVESTART ():

it starts the timer at microwave mode and gives the option to stop or pause the mode, there is a one that works the same way for the grill mode.

GRILLIDLE ():

it gives us the choice to either start or change the timer value and a similar mode exists for the microwave.

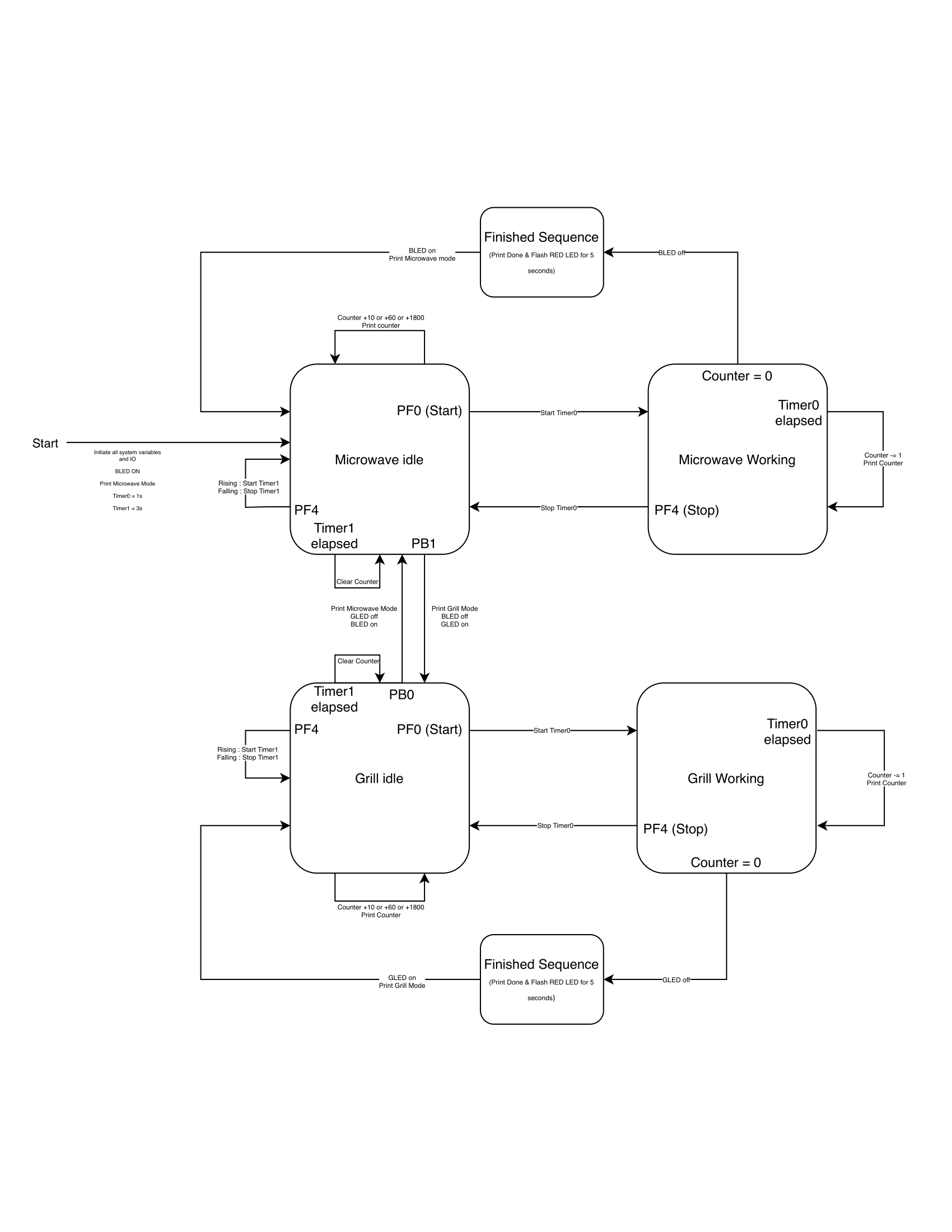
WELCOME ():

this function run at start and the microwave idle is set as the default mode with 100 seconds default timer.

Keypad\_scanner ():

It uses if loop as the keypad need multiplexing to obtain which button is clicked.

Finite state machine diagram



Non-functional merits

STACK USAGE

Category Max Use Total Use

------------------------ ------- ---------

interrupt 0 0

Program entry 96 96

Uncalled function 120 504

7'478 bytes of readonly code memory

634 bytes of readonly data memory

5'745 bytes of readwrite data memory

Ram

As we do not utilize the heap, so the stack is equal to the ram

Rom

0x1f68 = 8Kb

CPU utilization

We made a macro function to use with debugging at breakpoints to use the systick timer current register to obtain the value of the current at the beginning and end of function ans subtract the to get the total number of cycles and the divide it by the clock frequency.

Initialization + Welcome: 207 ms

Keypad interrupt: 100ms

UART Interrupt Stop: 169 ms

UART Interrupt Start: 44 ms

UART Interrupt seconds set: 80 us

UART Keypad Interrupt for starting any mode: 100ms

UART Keypad Interrupt for stopping any mode: 100ms

Power

P= V\*I

Since it is an estimate and at runtime the current does not change as the clock frequency does not change, so we get the run time current from the

datasheet = 17.6 , and sleep current from datasheet = 11.8 at all peripherals on , so that’s not the most accurate assumption as not all peripherals are on and their usage change through the run time , yet its is the best estimation we can get . If we need the real runtime power, we will probably need to use an oscilloscope probe to get interrupts current as its time is so small, so a multi-meter will not be able to scan it , and since the runtime is so small compared to the sleep time , we can assume that the sleep time is the power consumption of the system.

Runtime Power = 3.3\*17.6= 58.08mW

Sleep Power =3.3\*11.8=38.94mW

Files submitted

main.c

gpio.h

gpio.c

tm4c123gh6pm.h

Final-Project-CSE318-ASU-CHEP.map

Finite State Machine Diagram.pdf

Cycle.mac