Lab 11 - MotorHandForce

Table of motor speed values (duty cycle amount)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Motor speed name | ADC value (bottom threshold) (hex) | Motor State Value (raw) | PWM Pulse Width Settings (hex) | Pulse Width (ms) |
| STOP | N/A | 192 | C0 | 1.5 |
| REALLY\_REALLY\_SLOW | 0300 | 190 | BE | 1.48 |
| REALLY\_SLOW | 0700 | 186 | BA | 1.45 |
| QUITE\_SLOW | 1000 | 180 | B4 | 1.4 |
| MEDIUM | 1200 | 170 | AA | 1.33 |
| FAST | 1400 | 128 (max) | 80 | 1 |

Code:

//////////////////////////////

//Willard Wider

//07/01/18

//ELEC3800

//Lab11

//Analog control of PWM motor

//////////////////////////////

/\* DriverLib Includes \*/

**#include** <ti/devices/msp432p4xx/driverlib/driverlib.h>

/\* Standard Includes \*/

**#include** <stdint.h>

**#include** <stdbool.h>

**#include** <stdio.h>

//pre-defined timer values for each speed (all CW)

**#define** REAL\_STOP 0//let's be real. don't have to calibrate for middle if you don't send a signal #rollSafe

**#define** STOP 192//7.5% duty cycle -> 1.5ms - theoretical stop

**#define** REALLY\_REALLY\_SLOW 190//

**#define** REALLY\_SLOW 186//

**#define** QUITE\_SLOW 180//

**#define** MEDIUM 170//

**#define** FAST 128//5% duty cycle -> 1ms

**#define** PERIOD 2560//period of "carrier wave"

//2560 = 20ms

//pre-defined analog values for speeds

//ADC for 3.3 volts goes from 0 to 3FFF

**#define** SPEED\_1 0x0300//slowest

**#define** SPEED\_2 0x0700//

**#define** SPEED\_3 0x1000//

**#define** SPEED\_4 0x1200//

**#define** SPEED\_5 0x1400//fastest

//statics

**static** **volatile** uint16\_t curADCResult = 0x0000;

**static** **volatile** **long** avg100Results = 0;

**static** **volatile** **int** counter = 0;

**static** uint\_fast16\_t newPWMMode = STOP;

uint\_fast16\_t **GetNewPWMSpeed**();

/\* Timer\_A PWM Configuration Parameter \*/

Timer\_A\_PWMConfig pwmConfig =

{

TIMER\_A\_CLOCKSOURCE\_SMCLK,

TIMER\_A\_CLOCKSOURCE\_DIVIDER\_1,

PERIOD,

TIMER\_A\_CAPTURECOMPARE\_REGISTER\_1,

TIMER\_A\_OUTPUTMODE\_RESET\_SET,

STOP//stop it first off

};

**int** **main**(**void**)

{

**printf**("Halting watchdog...\n");

/\* Halting the watchdog \*/

MAP\_WDT\_A\_holdTimer();

**printf**("Compleet, init PWM...\n");

//PWM\_CODE/////////////////////////////////////////////////////////////////////////////////////////////////

/\* Setting MCLK to REFO at 128Khz for LF mode Setting SMCLK to 64Khz \*/

MAP\_CS\_setReferenceOscillatorFrequency(CS\_REFO\_128KHZ);

MAP\_CS\_initClockSignal(CS\_MCLK, CS\_REFOCLK\_SELECT, CS\_CLOCK\_DIVIDER\_1);

MAP\_CS\_initClockSignal(CS\_SMCLK, CS\_REFOCLK\_SELECT, CS\_CLOCK\_DIVIDER\_1);

MAP\_PCM\_setPowerState(PCM\_AM\_LF\_VCORE0);

/\* Configuring GPIO2.4 as peripheral output for PWM \*/

MAP\_GPIO\_setAsPeripheralModuleFunctionOutputPin(GPIO\_PORT\_P2, GPIO\_PIN4, GPIO\_PRIMARY\_MODULE\_FUNCTION);

/\* Configuring Timer\_A \*/

MAP\_Timer\_A\_generatePWM(TIMER\_A0\_BASE, &pwmConfig);

//!PWM\_CODE////////////////////////////////////////////////////////////////////////////////////////////////

**printf**("Complete, init ADC...\n");

//ADC\_CODE/////////////////////////////////////////////////////////////////////////////////////////////////

/\* Setting Flash wait state \*/

MAP\_FlashCtl\_setWaitState(FLASH\_BANK0, 1);

/\* Setting DCO to 48MHz \*/

MAP\_CS\_setDCOCenteredFrequency(CS\_DCO\_FREQUENCY\_48);

//![Single Sample Mode Configure]

/\* Initializing ADC (MCLK/1/4) \*/

MAP\_ADC14\_enableModule();

MAP\_ADC14\_initModule(ADC\_CLOCKSOURCE\_MCLK, ADC\_PREDIVIDER\_1, ADC\_DIVIDER\_4, 0);//needed now

/\* Configuring GPIOs (5.5 A0) \*/

//now it's 4.0

MAP\_GPIO\_setAsInputPin(GPIO\_PORT\_P4,GPIO\_PIN0);//clears pull up resistors i guess

MAP\_GPIO\_setAsPeripheralModuleFunctionInputPin(GPIO\_PORT\_P4, GPIO\_PIN0, GPIO\_TERTIARY\_MODULE\_FUNCTION);

/\* Configuring ADC Memory \*/

MAP\_ADC14\_configureSingleSampleMode(ADC\_MEM0, true);

MAP\_ADC14\_configureConversionMemory(ADC\_MEM0, ADC\_VREFPOS\_AVCC\_VREFNEG\_VSS, ADC\_INPUT\_A13, false);

/\* Configuring Sample Timer \*/

MAP\_ADC14\_enableSampleTimer(ADC\_MANUAL\_ITERATION);

/\* Enabling/Toggling Conversion \*/

MAP\_ADC14\_enableConversion();

MAP\_ADC14\_toggleConversionTrigger();

//![Single Sample Mode Configure]

/\* Enabling interrupts \*/

MAP\_ADC14\_enableInterrupt(ADC\_INT0);

MAP\_Interrupt\_enableInterrupt(INT\_ADC14);

MAP\_Interrupt\_enableMaster();

//!ADC\_CODE////////////////////////////////////////////////////////////////////////////////////////////////

**printf**("Complete, starting loop\n");

**while** (1)

{

newPWMMode = GetNewPWMSpeed();

**if**(newPWMMode != pwmConfig.dutyCycle)

{

//update the value

**printf**("Updating PWM value to %d\n",newPWMMode);

pwmConfig.dutyCycle = newPWMMode;

MAP\_Timer\_A\_generatePWM(TIMER\_A0\_BASE, &pwmConfig);

}

}

}

//![Single Sample Result]

/\* ADC Interrupt Handler. This handler is called whenever there is a conversion

\* that is finished for ADC\_MEM0.

\*/

**void** **ADC14\_IRQHandler**(**void**)

{

uint64\_t status = MAP\_ADC14\_getEnabledInterruptStatus();

MAP\_ADC14\_clearInterruptFlag(status);

**if** (ADC\_INT0 & status)

{

//curADCResult = MAP\_ADC14\_getResult(ADC\_MEM0);

avg100Results+=MAP\_ADC14\_getResult(ADC\_MEM0);

counter++;

**if**(counter >= 100)

{

curADCResult = avg100Results / 100;

avg100Results = 0;

counter=0;

//for debugging

**if**(0)

{

**printf**("curADCResult=%x\n",curADCResult);

}

}

MAP\_ADC14\_toggleConversionTrigger();

}

}

//![Single Sample Result]

uint\_fast16\_t **GetNewPWMSpeed**()

{

//since it's cascading if statements, check ADC level from low to high

**if**(curADCResult <= SPEED\_1)

{

**return** REAL\_STOP;

}

**else** **if** (curADCResult <= SPEED\_2)

{

//means it's between speed 1 and speed 2, etc.

**return** REALLY\_REALLY\_SLOW;

}

**else** **if** (curADCResult <= SPEED\_3)

{

**return** REALLY\_SLOW;

}

**else** **if** (curADCResult <= SPEED\_4)

{

**return** QUITE\_SLOW;

}

**else** **if** (curADCResult <= SPEED\_5)

{

**return** MEDIUM;

}

**else**//it's above SPEED\_5

{

**return** FAST;

}

}