Mechtron 3TA4 Lab 5 Report

Ho (Brian) Chiu

McMaster University

MacID: chiuh1

Student #: 400054774

1. Angular Resolution of a 48 step/revolution stepper motor
2. 400054774, 74 > 66 so my period will be 74 - 33 = 41
3. a) Half step:

Motor will step steps,

b) Full step:

1. For a 4MHz clock, prescaler = 399 for both modes. This makes the clock cycle at 10kHz
2. Half step: ARR = 4270
3. Full step: ARR = 8541
4. C code for timer and clock

TIM3 config fucntion generated by cubeMX: Period set to 8541 to start in Full step

static void MX\_TIM3\_Init(void)

{

/\* USER CODE BEGIN TIM3\_Init 0 \*/

/\* USER CODE END TIM3\_Init 0 \*/

TIM\_ClockConfigTypeDef sClockSourceConfig = {0};

TIM\_MasterConfigTypeDef sMasterConfig = {0};

/\* USER CODE BEGIN TIM3\_Init 1 \*/

/\* USER CODE END TIM3\_Init 1 \*/

htim3.Instance = TIM3;

htim3.Init.Prescaler = 399;

htim3.Init.CounterMode = TIM\_COUNTERMODE\_UP;

htim3.Init.Period = 8541;

htim3.Init.ClockDivision = TIM\_CLOCKDIVISION\_DIV1;

htim3.Init.AutoReloadPreload = TIM\_AUTORELOAD\_PRELOAD\_ENABLE;

if (HAL\_TIM\_Base\_Init(&htim3) != HAL\_OK)

{

Error\_Handler();

}

sClockSourceConfig.ClockSource = TIM\_CLOCKSOURCE\_INTERNAL;

if (HAL\_TIM\_ConfigClockSource(&htim3, &sClockSourceConfig) != HAL\_OK)

{

Error\_Handler();

}

sMasterConfig.MasterOutputTrigger = TIM\_TRGO\_RESET;

sMasterConfig.MasterSlaveMode = TIM\_MASTERSLAVEMODE\_DISABLE;

if (HAL\_TIMEx\_MasterConfigSynchronization(&htim3, &sMasterConfig) != HAL\_OK)

{

Error\_Handler();

}

/\* USER CODE BEGIN TIM3\_Init 2 \*/

if(HAL\_TIM\_Base\_Start\_IT(&htim3) != HAL\_OK) {

//Starting Error

Error\_Handler();

}

/\* USER CODE END TIM3\_Init 2 \*/

}

TIM3 Interrupt function: Increment the phase\_counter in the correct direction

Full step has 4 phases, half step has 8

void HAL\_TIM\_PeriodElapsedCallback(TIM\_HandleTypeDef \*htim){

if ((\*htim).Instance==TIM3) { //tim3 clock

switch(STATE){

case CW:

phase\_count = (phase\_count+1)%4; //count forwards

cw\_or\_ccw\_state();

break;

case CW\_HALF:

phase\_count = (phase\_count+1)%8;

cw\_half\_or\_ccw\_half\_state();

break;

case CCW:

phase\_count--; //count backwards

if(phase\_count>3) phase\_count = 3;

cw\_or\_ccw\_state();

break;

case CCW\_HALF:

phase\_count--; //count backwards

if(phase\_count>7) phase\_count = 7;

cw\_half\_or\_ccw\_half\_state();

break;

}

sprintf(lcd\_buffer,"%u",phase\_count+1);

BSP\_LCD\_GLASS\_Clear();

BSP\_LCD\_GLASS\_DisplayString((uint8\_t\*)lcd\_buffer); //display phase

}

}

Button press interrupt function: Select toggles direction, left toggles full or half step, up and down change speed.

void HAL\_GPIO\_EXTI\_Callback(uint16\_t GPIO\_Pin) { //buffer button pressed

switch(GPIO\_Pin) {

case GPIO\_PIN\_0: //SELECT

if(STATE==CW) STATE = CCW;

else if(STATE==CCW) STATE = CW;

else if(STATE==CW\_HALF) STATE = CCW\_HALF;

else if(STATE==CCW\_HALF) STATE = CW\_HALF;

break;

case GPIO\_PIN\_1: //LEFT

if(STATE==CW) STATE = CW\_HALF;

else if(STATE==CW\_HALF) STATE = CW;

else if(STATE==CCW) STATE = CCW\_HALF;

else if(STATE==CCW\_HALF) STATE = CCW;

if(STATE == CW || STATE == CCW) htim3.Init.Period \*= 2; //state changed to CW or CCW, double the period

else htim3.Init.Period /= 2; //state changed to CW\_HALF or CCW\_HALF, half the period

if (HAL\_TIM\_Base\_Init(&htim3) != HAL\_OK) Error\_Handler();

break;

case GPIO\_PIN\_2: //RIGHT

break;

case GPIO\_PIN\_3: //UP

if(htim3.Init.Period > 1000) htim3.Init.Period -= 1000; //speed up, decrease period

if (HAL\_TIM\_Base\_Init(&htim3) != HAL\_OK) Error\_Handler();

break;

case GPIO\_PIN\_5: //DOWN

htim3.Init.Period += 1000; //decrease speed, increase period

if (HAL\_TIM\_Base\_Init(&htim3) != HAL\_OK) Error\_Handler();

break;

}

}

Full step mode output pin states:

void cw\_or\_ccw\_state(void){ //pin order for CW or CCW

switch(phase\_count){

case 0: //1

HAL\_GPIO\_WritePin(GPIOE,OUT\_A\_Pin,GPIO\_PIN\_SET);

HAL\_GPIO\_WritePin(GPIOE,OUT\_B\_Pin,GPIO\_PIN\_RESET);

HAL\_GPIO\_WritePin(GPIOE,OUT\_C\_Pin,GPIO\_PIN\_SET);

HAL\_GPIO\_WritePin(GPIOE,OUT\_D\_Pin,GPIO\_PIN\_RESET);

break;

case 1: //2

HAL\_GPIO\_WritePin(GPIOE,OUT\_A\_Pin,GPIO\_PIN\_SET);

HAL\_GPIO\_WritePin(GPIOE,OUT\_B\_Pin,GPIO\_PIN\_RESET);

HAL\_GPIO\_WritePin(GPIOE,OUT\_C\_Pin,GPIO\_PIN\_RESET);

HAL\_GPIO\_WritePin(GPIOE,OUT\_D\_Pin,GPIO\_PIN\_SET);

break;

case 2: //3

HAL\_GPIO\_WritePin(GPIOE,OUT\_A\_Pin,GPIO\_PIN\_RESET);

HAL\_GPIO\_WritePin(GPIOE,OUT\_B\_Pin,GPIO\_PIN\_SET);

HAL\_GPIO\_WritePin(GPIOE,OUT\_C\_Pin,GPIO\_PIN\_RESET);

HAL\_GPIO\_WritePin(GPIOE,OUT\_D\_Pin,GPIO\_PIN\_SET);

break;

case 3: //4

HAL\_GPIO\_WritePin(GPIOE,OUT\_A\_Pin,GPIO\_PIN\_RESET);

HAL\_GPIO\_WritePin(GPIOE,OUT\_B\_Pin,GPIO\_PIN\_SET);

HAL\_GPIO\_WritePin(GPIOE,OUT\_C\_Pin,GPIO\_PIN\_SET);

HAL\_GPIO\_WritePin(GPIOE,OUT\_D\_Pin,GPIO\_PIN\_RESET);

break;

}

}

Half step mode output pin states:

void cw\_half\_or\_ccw\_half\_state(void){ //pin order for CW\_HALF or CCW\_HALF

switch(phase\_count){

case 0: //1

HAL\_GPIO\_WritePin(GPIOE,OUT\_A\_Pin,GPIO\_PIN\_SET);

HAL\_GPIO\_WritePin(GPIOE,OUT\_B\_Pin,GPIO\_PIN\_RESET);

HAL\_GPIO\_WritePin(GPIOE,OUT\_C\_Pin,GPIO\_PIN\_SET); HAL\_GPIO\_WritePin(GPIOE,OUT\_D\_Pin,GPIO\_PIN\_RESET);

break;

case 1: //2

HAL\_GPIO\_WritePin(GPIOE,OUT\_A\_Pin,GPIO\_PIN\_SET);

HAL\_GPIO\_WritePin(GPIOE,OUT\_B\_Pin,GPIO\_PIN\_RESET) HAL\_GPIO\_WritePin(GPIOE,OUT\_C\_Pin,GPIO\_PIN\_RESET);

HAL\_GPIO\_WritePin(GPIOE,OUT\_D\_Pin,GPIO\_PIN\_RESET);

break;

case 2: //3

HAL\_GPIO\_WritePin(GPIOE,OUT\_A\_Pin,GPIO\_PIN\_SET);

HAL\_GPIO\_WritePin(GPIOE,OUT\_B\_Pin,GPIO\_PIN\_RESET); HAL\_GPIO\_WritePin(GPIOE,OUT\_C\_Pin,GPIO\_PIN\_RESET); HAL\_GPIO\_WritePin(GPIOE,OUT\_D\_Pin,GPIO\_PIN\_SET);

break;

case 3: //4

HAL\_GPIO\_WritePin(GPIOE,OUT\_A\_Pin,GPIO\_PIN\_RESET); HAL\_GPIO\_WritePin(GPIOE,OUT\_B\_Pin,GPIO\_PIN\_RESET);

HAL\_GPIO\_WritePin(GPIOE,OUT\_C\_Pin,GPIO\_PIN\_RESET);

HAL\_GPIO\_WritePin(GPIOE,OUT\_D\_Pin,GPIO\_PIN\_SET);

break;

case 4: //5 HAL\_GPIO\_WritePin(GPIOE,OUT\_A\_Pin,GPIO\_PIN\_RESET); HAL\_GPIO\_WritePin(GPIOE,OUT\_B\_Pin,GPIO\_PIN\_SET); HAL\_GPIO\_WritePin(GPIOE,OUT\_C\_Pin,GPIO\_PIN\_RESET);

HAL\_GPIO\_WritePin(GPIOE,OUT\_D\_Pin,GPIO\_PIN\_SET);

break;

case 5: //6

HAL\_GPIO\_WritePin(GPIOE,OUT\_A\_Pin,GPIO\_PIN\_RESET); HAL\_GPIO\_WritePin(GPIOE,OUT\_B\_Pin,GPIO\_PIN\_SET); HAL\_GPIO\_WritePin(GPIOE,OUT\_C\_Pin,GPIO\_PIN\_RESET);

HAL\_GPIO\_WritePin(GPIOE,OUT\_D\_Pin,GPIO\_PIN\_RESET);

break;

case 6: //7

HAL\_GPIO\_WritePin(GPIOE,OUT\_A\_Pin,GPIO\_PIN\_RESET);

HAL\_GPIO\_WritePin(GPIOE,OUT\_B\_Pin,GPIO\_PIN\_SET);

HAL\_GPIO\_WritePin(GPIOE,OUT\_C\_Pin,GPIO\_PIN\_SET);

HAL\_GPIO\_WritePin(GPIOE,OUT\_D\_Pin,GPIO\_PIN\_RESET);

break;

case 7: //8

HAL\_GPIO\_WritePin(GPIOE,OUT\_A\_Pin,GPIO\_PIN\_RESET);

HAL\_GPIO\_WritePin(GPIOE,OUT\_B\_Pin,GPIO\_PIN\_RESET); HAL\_GPIO\_WritePin(GPIOE,OUT\_C\_Pin,GPIO\_PIN\_SET);

HAL\_GPIO\_WritePin(GPIOE,OUT\_D\_Pin,GPIO\_PIN\_RESET);

break;

}

}