## COE 538 - Robot Guidance Challenge Final Report

Lab Section	02

Student Name	Student Number		
Jonathan Ma	500837227		
Stanley Tan	501017455		

## **Code Explanation**

	ORG	\$4000			
Entry:					
_Startup:	LDS	#\$4000			
	JSR JSR JSR JSR	initPORTS initAD initLCD clrLCD			
	JSR	initTCNT			
	CLI LDX JSR	#msg1 putsLCD			
	LDAA JSR LDX JSR	#\$8A cmd2LCD #msg2 putsLCD			
	LDAA JSR LDX JSR	#\$CO cmd2LCD #msg3 putsLCD			
	LDAA JSR LDX JSR	#\$C7 cmd2LCD #msg4 putsLCD			
main	JSR JSR LDAA JSR BRA	UPDT_READING UPDT_DISPL CRNT_STATE DISPATCHER main			

This is the startup code and the main code. The startup code initializes several functions which the eebot requires. The main code runs the same 4 lines repeatedly to execute subsections which carry out the code.

			FWD_ST	PULD BRSET LDAA STAA JSR MOVB JMP	PORTADO, \$04, NO_FWD_BUMP SEC_PTH_INT NEXT_D INIT_REV #REV_CRNT_STATE FWD_EXIT
			NO_FWD_BUMP	BRSET JMP MOVB JMP	PORTADO,\$08,NO_REV_BUMP INIT_BK_TRK #BK_TRK,CRNT_STATE FWD_EXIT
			NO_REV_BUMP	LDAA BEQ LDAA PSHA LDAA STAA	SENS_C NO_RT_INTXN NEXT_D PRI_PTH_INT NEXT_D
DISPATCHER	CMPA BNE JSR	#START NOT_START START_ST	NO_RT_INTXN	JSR MOVB JMP	INIT_RT_TRN #RT_TRN,CRNT_STATE FWD_EXIT
NOT_START	RTS CMPA BNE JMP	#FWD NOT_FORWARD FWD_ST	NO_RI_INIAN	BEQ LDAA BEQ LDAA PSHA LDAA	SENS B NO_LT_INTXN SENS A LT_TURN NEXT_D PRI_PTH_INT
NOT_FORWARD	CMPA BNE JSR RTS	#RT_TRN NOT_RT_TRN RT_TRN_ST	LT_TURN	STAA BRA LDAA PSHA LDAA STAA JSR	NEXT_D NO_SHFT_LT NEXT_D SEC_PTH_INT NEXT_D INIT_LT_TRN
NOT_RT_TRN	CMPA BNE JSR RTS	#LT_TRN NOT_LT_TRN LT_TRN_ST	NO_LT_INTXN	MOVB JMP LDAA BEQ JSR LDD	#IT_TRN_CRNT_STATE FWD_EXIT  SENS_C NO_SHET_RT PORTON COUNT2
NOT_LT_TRN	CMPA BNE JSR RTS	#REV NOT_REVERSE REV_ST	RT_FWD_DIS	CPD BLO JSR JMP	#INC_DIS RT_FWD_DIS INTT_FWD FWD_EXIT
NOT_REVERSE	CMPA BNE JMP	#BK_TRK NOT_BK_TRK BK_TRK_ST	NO_SHFT_RT	LDAA BEQ JSR LDD CPD BLO JSR	SENS C NO_SHFT_LT STÄRON COUNT1 #INC_DIS LT_FUD_DIS INIT_FWD
NOT_BK_TRK	CMPA BNE JSR RTS	#SBY NOT_SBY SBY_ST	NO_SHFT_LT FWD_STR_DIS	JMP JSR JSR LDD CPD	FWD_EXIT STARON FORTON COUNT1 #FWD_DIS
NOT_SBY DISP_EXIT	NOP RTS		FWD_EXIT	BLO JSR JMP	FWD_STR_DIS INIT_FWD main

This is the Dispatcher code. This helps the eebot identify which process to run next.

The code beside the Dispatcher tells the eebot to follow the black line which alternates between different states of rotation.

START_ST	JSR MOVB	PORTADO,\$04,NO_FWD INIT_FWD #FWD,CRNT_STATE START_EXIT
NO_FWD START_EXIT	NOP RTS	

This is the start state. When the front bumper is pressed, the bot will begin to start and run.

REV_ST	LDD	COUNT1					
	CPD BLO JSR LDD STD	#REV_DIS REV_ST STARFWD #0 COUNT1		ŔT_TRN_S	Γ	LDD CPD BLO JSR LDD STD	COUNT2 #STR_DIS RT_TRN_ST STAROFF #0 COUNT2
REV_U_TRN	LDD CPD BLO JSR LDAA BNE MOVB BRA	COUNT1 #UTRN_DIS REV_U_TRN INIT_FWD RETURN BK_TRK_REV #FWD,CRNT_STATE REV_EXIT		RT_TURN_	LOOP	LDD CPD BLO JSR LDAA BNE MOVB	COUNT2 #TRN_DIS RT_TURN_LOOP INIT_FWD RETURN BK_TRK_RT_TRN #FWD,CRNT_STATE
BK_TRK_REV	JSR MOVB	ÎNÎT_ÊWD #BK_TRK,CRNT_STATE		BK_TRK_R	r_trn	BRA MOVB	RT_TRN_EXTT
REV_EXIT	RTS			RT_TRN_E	KIT	RTS	
LT_TRN_ST	LDD CPD BLO JSR LDD STD	COUNT1 #STR_DIS LT_TRN_ST PORTOFF #0 COUNT1					
LT_TURN_LOOP	LDD CPD BLO JSR LDAA BNE MOVB BRA	COUNT1 #TRN_DIS LT_TURN_LOOP INIT_FWD RETURN BK_TRK_LT_TRN #FWD,CRNT_STATE LT_TRN_EXIT	BK_TR	_	JSR MOVB JMP	INIT_S	RNT_STATE _EXIT
BK_TRK_LT_TRN	MOVB	#BK_TRK,CRNT_STATE	MO_PV	_BOMP	LDAA BEQ BNE	REG_PA	THING
LT_TRN_EXIT	RTS				DNE	IRREG_	PATHING
REG_PATHING	LDAA BEQ PULA PULA STAA JSR MOVB JMP	SENS_C NO_RT_TRN NEXT_D INIT_RT_TRN #RT_TRN,CRNT_STATE BK_TRK_EXIT	IRREG	S_PATHING	LDA BEQ PUL STA JSR MOV JMP	NO_ A A NEX INI B #LT	LT_TRN
NO_RT_TRN  LEFT_TURN	LDAA BEQ LDAA BEQ PULA STAA PULA STAA PULA STAA JSR MOVB JMP	SENS_B RT_LINE_S SENS_A LEFT_TURN  NEXT_D NO_LINE_S  NEXT_D INIT_LT_TRN #LT_TRN, CRNT_STATE BK_TRK_EXIT	NO_LT	'_TRN '_TURN	LDA BEQ LDA BEQ PUL STA BRA PUL STA JSR MOV JMP	RT_A SEN RIG A NEX NO_A A NEX INI B #RT	LINE_S S_A HT_TURN T_D LINE_S

Have a subroutine which follows the lines, which depends on the sensor. These functions simply cause the eebot to turn based on a set of conditions which is called in the main code. One of the subroutines checks the state of the led and updates the Dispatcher.

The rest code consists of utility subroutine functions which have been covered in the previous labs and in the Guider code.

## **Problems Encountered**

Some of the main problems we encountered when working on the project is listed down below:

- Robot kept moving to the left
- One of the LED lights did not update/blink while the other ones did as expected
- Sometimes the LED stopped flashing after a while of running the bot

These are the main problems we encountered during our work on the project. We tried many different solutions to all these problems but none of them seemed to work. One of the possible scenarios was that we received a faulty bot where one of the sensors did not update or work as expected. Another problem could have been within the code portion of the project. Since the bot kept moving left and couldn't get a sense of following the black line, one possible explanation is that we coded the bot and it got stuck in a state of rotation. Evidently, with the bot not being able to switch states based off of the sensors, it stayed in rotation and couldn't respond to any stimulation.

## **Key Takeaways**

Some key takeaways from this project we had is to have better time management of the software development and testing part of the project. Preparing different code blocks for our project ahead of time can allow us to test them altogether and figure out which ones are the most suitable for us. On top of that, we should have tested the code out more frequently instead of cramming it all in the last couple of days before our demo time. Another key takeaway is to be open to re-coding functions of previous labs. We had a mindset of keeping all the functions and code of the previous labs because they were running properly. As a result, we did not want to alter any lab 5 code until we kept troubleshooting the project and had no option but to change a few parts of that code. In hindsight, we should have been more open to looking at all possibilities of code alteration during the development and testing phase. This project was a great experience for the both of us, we learned a lot in terms of what it takes to properly manage and develop a software project. In the future, we would definitely start our work a few weeks prior to the due date and consistently test and debug our software while documenting both the bad and good parts of the code.