Date: 02/20/16

To: CSS 422, Winter 2016

From: Easy Riders

Subject: Team Progress Report 2

This week focused on solidifying details relating to fundamentals of how the disassembler project was all going to fit and work together. Thomas investigated structure of how machine code will be parsed and decoded. Ross continued to flesh out the disassembler prototype to incorporate parsing of opcodes more complex than NOP.

Status Report:

- Need full project timeline
    - Status: Pushed back to Sprint 3
- Transfer completed layouts to project source code
    - Notes: Due 2/19
    - Assigned: Thomas 2/16
    - Status:
        - Layouts from Spring 1 are transferred in. Framework of project code is now in place. td 2/16
- Begin formal coding for project
    - Notes: Build disassembler based off of input test code modules after framework is created.
    - Blockers: Requires transfer of existing layouts to be complete. Done. td 2/16
    - Status:
        - Wrote TestMOVE.X86 which tests the entirety of the MOVE opcode. All EA modes are tested for source and destination. Implementing disassembly of this test module will create most of the structure for the entire project. td 2/16
- Project Team Report 2
    - Notes: Due Saturday 2/20
    - Assigned: Thomas 2/19
    - Status:
        - Updated changes to Branch List table. td 2/19
        - Draft complete. td 2/20
- Continue creating branch, opcode, and subroutine logic layouts
    - Notes: Develop until basic idea of program flow is attained.
    - Assigned: Thomas 2/13
    - Status:
        - Removed Branch5 from Root Branch. td 2/19
        - Restructured Branch0. All paths complete. td 2/19
        - Restructured Branch4. All paths complete. td 2/19

- Updating naming for Branch4E7. All paths complete. td 2/19
- Completed all paths for Branch6. td 2/19
- Completed all paths for Branch8. td 2/19
- Completed all paths for Branch9. td 2/19
- Completed all paths for BranchB. td 2/19
- Completed all paths for BranchC. td 2/19
- Completed all paths for BranchD. td 2/19
- Completed all paths for BranchE. All Branch layouts complete, and all opcodes can be uniquely identified. td 2/19.
- NOP opcode flowchart done. td 2/19
- CLR opcode flowchart done. td 2/19
- Drafted flowcharts for decoding size, registers, and effective addresses. td 2/19

Deliverables:

Changes in the assigned opcodes required an audit of the previously constructed branch list table to verify which codes were removed and which ones had been added. The changes are reflected in the attached table. Opcodes no longer required or that were redundant analysis are grayed out.

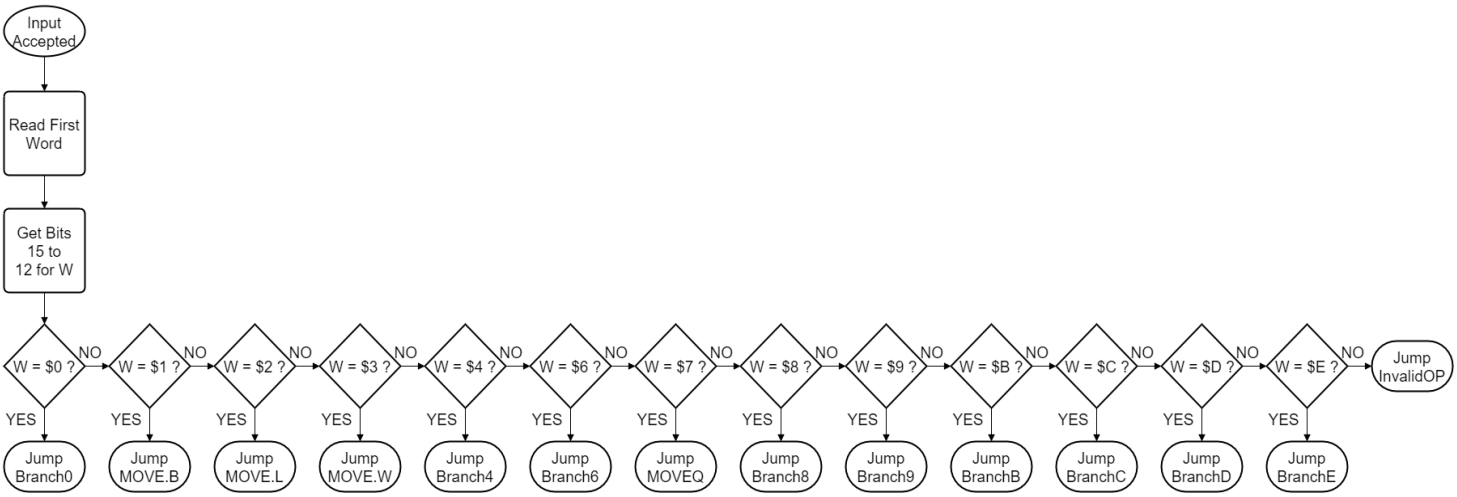
						Single Effective Address Operation Word Format														Brief Extension Word Format														Full Extension Word Format													
Mnemonic	Page	$W	$X	$Y	$Z	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ANDI	4 - 18	0	0			0	0	0	0	0	0	0	0	Size		EA Mode			EA Register			16-Bit Word Data										8-Bit Byte Data								32-Bit Long Data													
ADDI	4 - 9	0	3			0	0	0	0	0	0	1	1	Size		EA Mode			EA Register			16-Bit Word Data										8-Bit Byte Data								32-Bit Long Data													
BCHG	4 - 27	0				0	0	0	0	Register				1	0	1		EA Mode		EA Register																															
BCHG	4 - 27	0	8			0	0	0	0	1	0	0	0	0	1	EA Mode			EA Register			0	0	0	0	0	0	0	0	Bit Number																							
CMPI	4 - 79	0	C			0	0	0	0	1	1	0	0	Size		EA Mode			EA Register			16-Bit Word Data										8-Bit Byte Data								32-Bit Long Data													
EORI	4 - 102	0	A			0	0	0	0	1	0	1	0	Size		EA Mode			EA Register			16-Bit Word Data										8-Bit Byte Data								32-Bit Long Data													
SUBI	4 - 179	0	4			0	0	0	0	0	1	0	0	Size		EA Mode			EA Register			16-Bit Word Data										8-Bit Byte Data								32-Bit Long Data													
MOVE.B	4 - 116	1				0	0	0	1	Dest Register				Dest Mode		Src Mode			Src Register																																	
MOVE.L	4 - 116	2				0	0	1	0	Dest Register				Dest Mode		Src Mode			Src Register																																	
MOVE.W	4 - 116	3				0	0	1	1	Dest Register				Dest Mode		Src Mode			Src Register																																	
CLR	4 - 73	4	2			0	1	0	0	0	0	1	0	Size		EA Mode			EA Register																															
DIVS.L	4 - 92	4	C			0	1	0	0	1	1	0	0	0	1	EA Mode			EA Register	0	Register Dq			1	Size	0	0	0	0	0	0	0	Register Dr																			
NOP	4 - 147	4	E	7	1	0	1	0	0	1	1	1	0	0	1	1	1	0	0	0	1																													
RTS	4 - 169	4	E	7	5	0	1	0	0	1	1	1	0	0	1	1	1	0	1	0	1																													
JSR	4 - 109	4	E			0	1	0	0	1	1	1	0	1	0	EA Mode			EA Register																															
MOVEM	4 - 128	4				0	1	0	0	1	dir	0	0	1	size	EA Mode			EA Register	Register List Mask																															
LEA	4 - 110	4				0	1	0	0	Register				1	1	1		EA Mode		EA Register																															
MULS.L	4 - 135	4	C			0	1	0	0	1	1	0	0	0	0	EA Mode			EA Register	0	Register Di			1	Size	0	0	0	0	0	0	0	Register Dh																			
ADDQ	4 - 11	5				0	1	0	1	Data				0	Size	EA Mode			EA Register																															
BCC	4 - 25	6	4			0	1	1	0	0	1	0	0	8-Bit Displacement								16-Bit Displacement if 8-Bit Displacement = $00																																	
BGT	4 - 25	6	E			0	1	1	0	1	1	1	0	8-Bit Displacement								16-Bit Displacement if 8-Bit Displacement = $00																																	
BLE	4 - 25	6	F			0	1	1	0	1	1	1	1	8-Bit Displacement								16-Bit Displacement if 8-Bit Displacement = $00																																	
MOVEQ	4 - 134	7				0	1	1	1	Register				0	Data																																			
DIVS	4 - 92	8				1	0	0	0	Register				1	1	1	EA Mode		EA Register																																	
SUB	4 - 174	9				1	0	0	1	Register				Opmode			EA Mode		EA Register																																	
SUBA	4 - 177	9				1	0	0	1	Register				Opmode			EA Mode		EA Register																																	
CMP	4 - 75	B				1	0	1	1	Register				Opmode			EA Mode		EA Register																																	
CMPA	4 - 77	B				1	0	1	1	Register				Opmode			EA Mode		EA Register																																	
EOR	4 - 100	B				1	0	1	1	Register				Opmode			EA Mode		EA Register																																	
AND	4 - 15	C				1	1	0	0	Register				Opmode			EA Mode		EA Register																																	
MULS	4 - 135	C				1	1	0	0	Register				1	1	1	EA Mode		EA Register																																	
ADD	4 - 4	D				1	1	0	1	Register				Opmode			EA Mode		EA Register																																	
ADDA	4 - 7	D				1	1	0	1	Register				Opmode			EA Mode		EA Register																																	
ASR	4 - 21	E				1	1	1	0	Count/ Reg				0	Size		i/r	0	0	Register																																	
LSR	4 - 113	E				1	1	1	0	Count/ Reg				0	Size		i/r	0	1	Register																																	
ROR	4 - 161	E				1	1	1	0	Count/ Reg				0	Size		i/r	1	1	Register																																	
ASL	4 - 21	E				1	1	1	0	Count/ Reg				1	Size		i/r	0	0	Register																																	
LSL	4 - 113	E				1	1	1	0	Count/ Reg				1	Size		i/r	0	1	Register																																	
ROL	4 - 161	E				1	1	1	0	Count/ Reg				1	Size		i/r	1	1	Register																																	

All branching tree structure to uniquely identify all assigned opcodes is now complete.

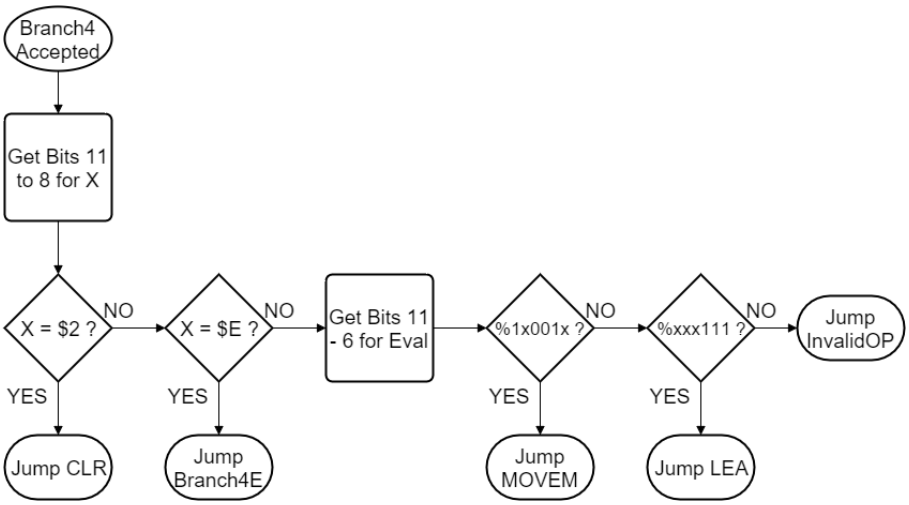
As a final experiment before regular development begins, an end-to-end experience for a simple opcode with effective addressing was created.

Read in code: 4200 (Expected: CLR.B (A1),D0)

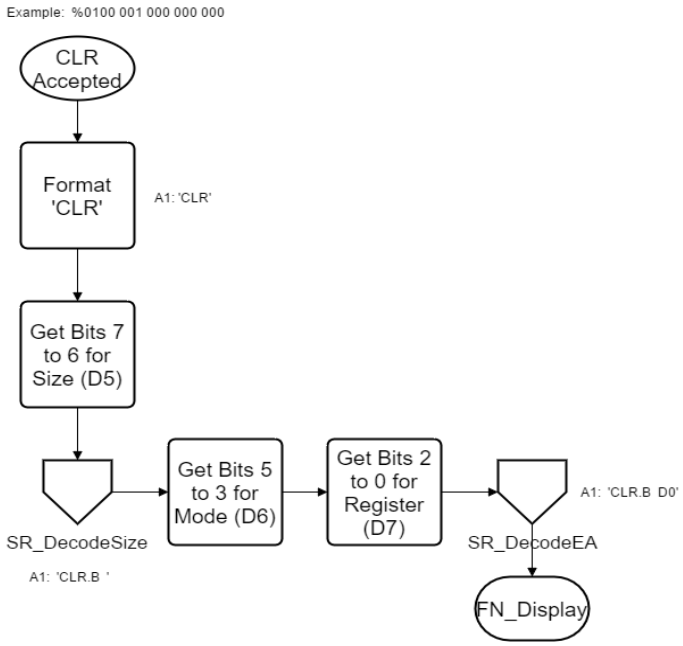
Branch on 4



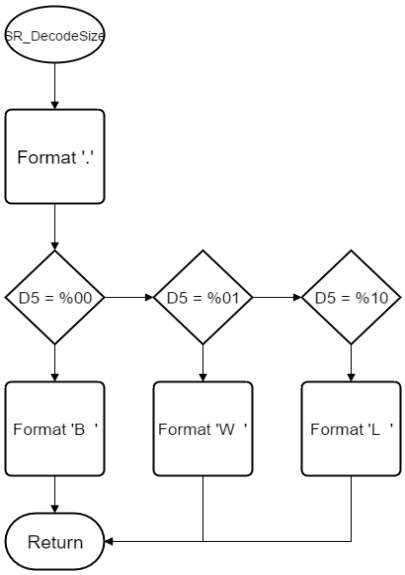
Branch on 2



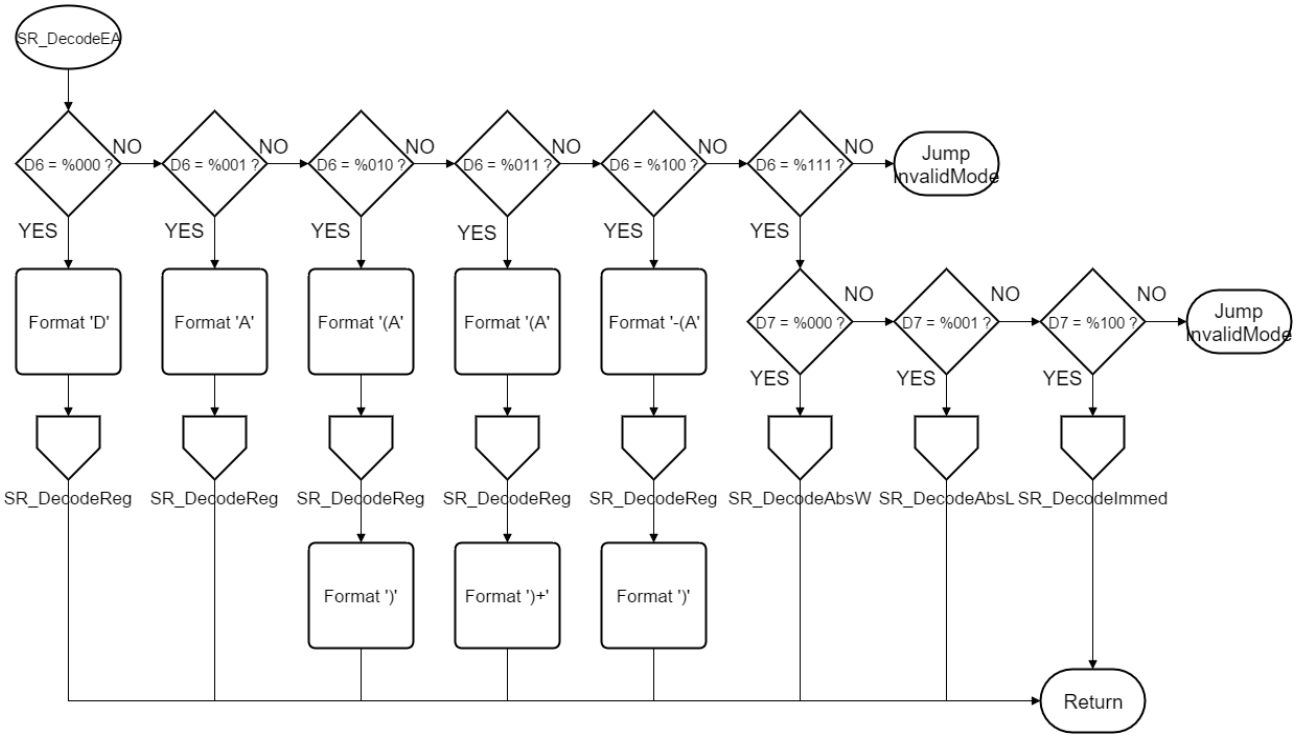
Branch on CLR



Format, process, then jump to subroutine SR_DecodeSize

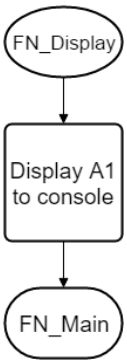


Return to CLR, format, then jump to subroutine SR_DecodeEA



Format, then jump to subroutine SR_DecodeReg

Format, then return to SR_DecodeReg, then return to CLR. Branch to function FN_Display



Join output buffers of memory address and assembler syntax (not shown) and print to console. Return to main function to process next command.

Problems:
While basic functionality is already implemented into the prototype, there are more complex tasks related to this project that still require experimentation before regular development can begin.

Work Scheduled:
Most tasks from Sprint 2 were completed as scheduled. We did not address the 'Need full project timeline' task. During Sprint 2, there was not enough functional knowledge of the program layout to properly schedule this, so it was not attempted.

Tasks for Sprint 3:

- Full audit of project specifications and deliverables
- Need full project timeline
- Built interactive client interface
- Implement minimum 5 new opcodes.

Self Evaluation:
We are currently on schedule with a solid foundation, but a steady development pace is required if we are to remain on track.