University College London

DEPARTMENT OF ELECTRONIC & ELECTRICAL ENGINEERING

Project Progress Report No. 2

Author:

Minduagas Jarmolovicius zceemja@ucl.ac.uk

Supervisor:
Prof. Robert Killey
r.killey@ucl.ac.uk

1 Progress

The following points have been done since last report:

- Upgraded assembler to support more complex operations, also changed syntax to comply with NASM;
- Upgraded automatisation using MakeFile;
- Implemented instruction memory using FPGA's M9K Memory;
- Have functional communication block, see Table 1;
- Implemented most of the instructions, see Table 2;

| Address | Function | Send | Return |
|---------|----------------------------|----------|------------------|
| 0x04 | Read UART0 flags | - | UART0 flags |
| 0x05 | Transmit to UART0 | TX byte | UART0 flags |
| 0x06 | Set DE0-Nano board LEDs | LED byte | - |
| 0x07 | Read DE0-Nano DIP switches | - | Lower DIP nibble |

Table 1: Addresses and functions for communication block

Project schedule as Grantt chart has been updated in the next page in table 3.

2 Difficulties encountered

Instruction memory (ROM) has been replaced with M9K memory instead of LC (logic cell), however, as this memory is clocked it caused further problems with program counter timings.

NASM-like assembler can have multiple very useful functions such as pre-compiler, macros, imports, db instruction (stores strings) etc. It is difficult to implement all these advanced functions.

Due to scale of project, byte order has been mixed (internally processor operates at little-endian, however addresses in instructions are written as big-endian), this needs to be sorted out.

3 Failure Risk Assessment

There are no updates on failure risk assessment. As before, the most dominant failure risk is running out of time project is behind schedule.

See table 3 for schedule. In 2 weeks is scheduled to start consider OISC architecture, however, the RISC processor is still far from completion. Benchmark development might need to extended to be completed during winter holidays. Higher level RISC compiler might be replaced by advanced functions in NASM-like compiler.

4 Updated Safety Risk Assessment

There are no updates on safety risk assessment.

5 Help and Advice Needed

At this state no help is needed, and any issues and advices are sorted out and discussed in weekly supervisor meetings.

| MOVE | Instr. | Description | Completed | | |
|--|-------------------------|---------------------------------------|-----------|--|--|
| ADD Arithmetical addition x SUB Arithmetical subtraction x AND Logical ADD x OR Logical AOR x XOR Logical XOR x MUL Arithmetical multiplication x BR Arithmetical division (inc. modulus) x BR Branch on registers equal x I register instructions SLL Shift left logical SRA Shift right arithmetical s SRA Shift right arithmetical signed x LWHI Load word (logical signed x SWIII Load word (low byte) x SWIII Store word (light byte) x SWIII Load word (low byte) x SWIO Store word (low byte, erg. only) x SWLO Store word (light byte reg. (only for MUL & DIV) x GETAII Get ALU high byte reg. (only for MUL & DIV) x GETIF Cet intercrupt flags x PUSH | 2 register instructions | | | | |
| SUB Arithmetical subtraction x AND Logical AND x OR Logical AND x NOR Logical AND x NOR Logical AND x NOR Logical AND x MUL Arithmetical multiplication x Intermedical division (inc. modulus) Branch on registers equal Intermedical division (inc. modulus) Intermedical divis | MOVE | | X | | |
| AND Logical AND x OR Logical OR x MUL Arithmetical division (inc. modulus) x DIV Arithmetical division (inc. modulus) x BR Branch on registers equal x I register instructions SLL Shift left logical SRA Shift right arithmetical signed x LWHI Load word (high byte) x SWHI Load word (lipit byte, reg. only) x LWLO Load word (low byte, stores high byte reg.) x SWLO Store word (low byte, stores high byte reg.) x INC Increase by 1 x GETAH Get ALU high byte reg. (only for MUL & DIV) x GETH Get interrupt flags x PUSH Push to stack x POP Pop from stack x COM Send/Receive to/from com. block x SETI Set immediate from register x BEQ Branch on greater than x <t< td=""><td></td><td></td><td>X</td></t<> | | | X | | |
| OR Logical OR x XOR Logical XOR x MUL. Arithmetical Multiplication x DIV Arithmetical division (inc. modulus) x BR Branch on registers equal x I register instructions SLL Shift left logical SRA Shift right arithmetical x SRAS Shift right arithmetical signed x LWIII Load word (high byte) x SWH0 Store word (lingh byte, reg. only) x SWLO Store word (low byte), stores high byte reg.) x INC Increase by 1 x GETAH Get ALU high byte reg. (only for MUL & DIV) x GETTIF Get interrupt flags x PUSH Push to stack x COM Send/Receive to/from com. block x SETI Set immediate from register x BEQ Branch on greater than x BGE Branch on greater equal than x BZ <td></td> <td>Arithmetical subtraction</td> <td>X</td> | | Arithmetical subtraction | X | | |
| XOR Logical XOR x MUL Arithmetical multiplication x DIV Arithmetical division (inc. modulus) x BR Branch on registers equal x Iregister instructions SIL Shift right logical SRA Shift right arithmetical signed x LWHI Load word (high byte) x SWHI Store word (high byte, reg. only) x LWLO Load word (low byte) x SWLO Store word (low byte, stores high byte reg.) x INC Increase by 1 x DEC Decrease by 1 x GETAH Get ALU high byte reg. (only for MUL & DIV) x GETIF Get interrupt flags x PUSH Push to stack x COM Send/Receive to/from com. block x SETI Set immediate from register x BEQ Branch on greater than x BGT Branch on greater equal than x BGT Branch on greater equal than x BG Branch on greater equal than x BC Branch on greater equal than x BC Ext ALU carry-in Set ALU carry- | | | X | | |
| MUL Arithmetical multiplication x DIV Arithmetical division (inc. modulus) x BR Branch on registers equal x I register instructions SIL Shift right logical SRA Shift right arithmetical x SRAS Shift right arithmetical signed x LWHI Load word (high byte, reg. only) x SWHI0 Store word (low byte) x SWLI0 Store word (low byte, stores high byte reg.) x INC Increase by 1 x DEC Decrease by 1 x GETAH Get ALU high byte reg. (only for MUL & DIV) x GETIF Get interrupt flags x PUSH Push to stack x POP Pop from stack x SETI Send/Receive to/from com. block x SETI Branch on equal x BGE Branch on greater equal than x BZ Branch on greater equal than x BZ Branch on greater equal than x BZ <t< td=""><td></td><td></td><td>X</td></t<> | | | X | | |
| DIV Branch on registers equal SLL Shift left logical SRL Shift right arithmetical signed LWHI Load word (high byte) x SWHO Store word (low byte) x SWLO Store word (low byte, stores high byte reg.) x SWLO Store word (low byte, stores high byte reg.) x SWLO Store word (low byte reg. (only for MUL & DIV) x GETH Get interrupt flags PUSH Push to stack x SETI Set immediate from register BEQ Branch on greater than BGE Branch on greater than BGE Branch on greater equal than BGE Branch on greater equal than BGE Branch on greater equal than SET Return from function therrupt SET SET Set ALU carry-in SET SET SET ALU sign SETS SET Set ALU sign SETS SET Set ALU sign SESETS Enable ALU negative SESETN Relative jump Relative jump Relative jump Relative jump **Teristractions** **Teristracti | | | X | | |
| Branch on registers equal | | <u> </u> | X | | |
| SLL Shift left logical SRL Shift right logical SRAS Shift right arithmetical SRAS Shift right arithmetical signed LWHI Load word (high byte) x SWHI Store word (high byte, reg. only) x LWLO Load word (low byte, reg. only) x LWLO Load word (low byte, stores high byte reg.) x SWHO Store word (low byte, stores high byte reg.) x INC Increase by 1 x GETAH Get ALU high byte reg. (only for MUL & DIV) x GETH Get ALU high byte reg. (only for MUL & DIV) x GETH Get interrupt flags PUSH Push to stack x COM Send/Receive to/from com. block x SETI Set immediate from register BEQ Branch on equal x BET Branch on greater than x BGE Branch on greater equal than x BGE Branch on greater equal than x BT Branch on greater than x BC Branch on greater equal than x BT Branch on greater span x BT Return from function x JUMP Jump to address x RET Return from interrupt CLC Clear ALU carry-in SETC Set ALU carry-in SETC Set ALU carry-in SETS Set ALU sign SETS Set ALU negative SETN Set ALU negative SETN Set ALU negative SETN Set ALU negative SETN Set ALU negative Relative jump | | | X | | |
| SLL Shift left logical SRA Shift right logical SRAS Shift right arithmetical SRAS Shift right arithmetical signed LWHI Load word (high byte) x SWHI Store word (high byte, reg. only) x LWLO Load word (low byte) x SWLO Store word (low byte, stores high byte reg.) x INC Increase by 1 x GETAH Get ALU high byte reg. (only for MUL & DIV) x GETHF Get interrupt flags PUSH Push to stack x POP Pop from stack x SETI Set immediate from register BEQ Branch on equal x BGT Branch on greater equal than BZ Branch on greater equal than BZ Branch on greater equal than BZ Branch on greater equal than SZ Return from function x JUMP Jump to address x ETI Return from function x JUMP SET Set ALU carry-in CLS Clear ALU carry-in CLS Clear ALU sign SETS SETS Enable ALU negative SETN SET Enable ALU negative SETN SET Enable ALU negative SESTN Enable ALU negative SESTN Felative jump | BR | | | | |
| SRL Shift right logical SRAS Shift right arithmetical SRAS Shift right arithmetical signed LWHI Load word (high byte) x SWHI Store word (high byte, reg. only) x LWLO Load word (low byte) x SWLO Store word (low byte) x SWLO Increase by 1 x DEC Decrease by 1 x DEC Decrease by 1 x GETIF Get interrupt flags PUSH Push to stack x POP Pop from stack x SETI Set immediate from register BEQ Branch on greater than x BC Branch on greater equal than BC Branch on greater equal than BC Branch on zero x CALL Call function, put return to stack x RET Return from function x JUMP Jump to address x RETI Set ALU carry-in SETS Set ALU sign SSETS Set ALU sign SSETS Enable ALU negative SSETN Set alve in a sign and in | CT T | - | | | |
| SRA Shift right arithmetical SRAS Shift right arithmetical signed LWHI Load word (high byte) x SWHI Store word (high byte, reg. only) x LWLO Load word (low byte) x SWLO Store word (low byte, stores high byte reg.) x INC Increase by 1 x GETAH Get ALU high byte reg. (only for MUL & DIV) x GETH Get interrupt flags PUSH Push to stack x POP Pop from stack x SETI Set immediate from register BEQ Branch on equal x BGE Branch on greater equal than x BZ Branch on zero x CALL Call function, put return to stack x RET Return from function x JUMP Jump to address x RETI Return from interrupt CLC Clear ALU carry-in SETS Set ALU carry-in CLIN Clear ALU sign SSETS Enable ALU negative SETN Set ALU negative SETN Set ALU negative SESETN Enable ALU negative RJUMP Relative jump Relative jump Relative jump Relative jump Relative jump Relative jump Relative jump Relative jump Relative jump SA X X X X X X X X X X X X X | | | | | |
| SRAS Shift right arithmetical signed LWHI Load word (high byte) X SWHI Store word (high byte, reg. only) X LWLO Load word (low byte) X SWHO Store word (low byte, stores high byte reg.) X SWHO Store word (low byte, stores high byte reg.) X INC Increase by 1 X GETAH Get ALU high byte reg. (only for MUL & DIV) X GETHH Get interrupt flags CETHH Push to stack X POP Pop from stack X POP Pop from stack X SETI Set immediate from register SEQ Branch on equal X BGT Branch on greater than X BGE Branch on greater equal than BZ Branch on zero X ETH STANCH STAN | | | | | |
| LWHI Load word (high byte) x SWHII Store word (high byte, reg. only) x LWLO Load word (low byte) x SWLO Store word (low byte, stores high byte reg.) x INC Increase by 1 x DEC Decrease by 1 x GETAH Get ALU high byte reg. (only for MUL & DIV) x GETIF Get interrupt flags x PUSH Push to stack x POP Pop from stack x COM Send/Receive to/from com. block x SETI Set immediate from register x BEQ Branch on greater than x BGE Branch on greater equal than x BZ Branch on zero x Urgister instructions CALL Call function, put return to stack x RET Return from function x JUMP Jump to address x RETI Return from interrupt C.C. CLS | | | | | |
| SWHI Store word (high byte, reg. only) x LWLO Load word (low byte) x SWLO Store word (low byte, stores high byte reg.) x SWLO Increase by 1 x DEC Decrease by 1 x GETAH Get ALU high byte reg. (only for MUL & DIV) x GETIF Get interrupt flags x PUSH Push to stack x POP Pop from stack x COM Send/Receive to/from com. block x SETI Set immediate from register x BEQ Branch on equal x BC Branch on greater than x BC Branch on greater equal than x BZ Branch on preater equal than x BZ Branch on preater equal than x BZ Branch on register x CALL Return from function x JUMP Jump to address x RETI Return from interrupt x CLS Clear ALU carry-in x SETS Set ALU sign< | | | | | |
| LWLO Load word (low byte) x SWLO Store word (low byte, stores high byte reg.) x INC Increase by 1 x DEC Decrease by 1 x GETAH Get ALU high byte reg. (only for MUL & DIV) x GETIF Get interrupt flags x PUSH Push to stack x POP Pop from stack x COM Send/Receive to/from com. block x SETI Set immediate from register x BEQ Branch on equal x BGT Branch on greater than x BGE Branch on greater equal than x BZ Branch on zero x O register instructions CALL Call function, put return to stack x RET Return from function x JUMP Jump to address x RETI Return from interrupt Clear ALU carry-in CLS Clear ALU sign SETS SETS Set | | \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ | | | |
| SWLO Store word (low byte, stores high byte reg.) x INC Increase by 1 x DEC Decrease by 1 x GETAH Get ALU high byte reg. (only for MUL & DIV) x GETIF Get interrupt flags PUSH Push to stack x POP Pop from stack x COM Send/Receive to/from com. block x SETI Set immediate from register BEQ Branch on equal x BGT Branch on greater than x BGE Branch on greater equal than x BZ Branch on zero x CALL Call function, put return to stack x RET Return from function x JUMP Jump to address x RETI Return from interrupt CLC Clear ALU carry-in SETC Set ALU carry-in CLS Clear ALU sign SETS Set ALU sign SETS Set ALU negative SETN Enable ALU negative | | | | | |
| INC Increase by 1 x DEC Decrease by 1 x GETAH Get ALU high byte reg. (only for MUL & DIV) x GETIF Get interrupt flags x PUSH Push to stack x POP Pop from stack x COM Send/Receive to/from com. block x SETI Set immediate from register x BEQ Branch on equal x BGT Branch on greater than x BGE Branch on greater equal than x BZ Branch on zero x O register instructions CALL Call function, put return to stack x RET Return from function x JUMP Jump to address x RETI Return from interrupt ClC CLC Clear ALU carry-in Clear SET Set ALU sign SET SETS Set ALU negative SET SETN Enable ALU negative <td></td> <td></td> <td></td> | | | | | |
| DEC Decrease by 1 | | | | | |
| GETAH Get ALU high byte reg. (only for MUL & DIV) GETIF Get interrupt flags PUSH Push to stack POP Pop from stack COM Send/Receive to/from com. block SETI Set immediate from register BEQ Branch on equal BGT Branch on greater than BGE Branch on greater equal than BZ Branch on zero CALL Call function, put return to stack RET Return from function XX RET Return from function XX RETI Return from interrupt CLC Clear ALU carry-in SETC Set ALU sign SETS Set ALU sign SETS Set ALU sign SETS Enable ALU negative SETN Set ALU negative SETN Enable ALU negative SETN Relative jump Relative jump Relative jump Relative jum | | · · | | | |
| GETIF Get interrupt flags PUSH Push to stack | | | | | |
| PUSH Push to stack | | | X | | |
| POP Pop from stack x COM Send/Receive to/from com. block x SETI Set immediate from register BEQ Branch on equal x BGT Branch on greater than x BGE Branch on greater equal than x BZ Branch on zero x **CALL** Call function, put return to stack x RET Return from function x JUMP Jump to address x RETI Return from interrupt ** CLC Clear ALU carry-in ** SETC Set ALU sign ** SETS Set ALU sign ** SETS Enable ALU negative ** SETN Set ALU negative ** SETN Enable ALU negative ** RJUMP Relative jump ** | | | | | |
| COMSend/Receive to/from com. blockxSETISet immediate from registerBEQBranch on equalxBGTBranch on greater thanxBGEBranch on greater equal thanxBZBranch on zeroxCALLCall function, put return to stackxRETReturn from functionxJUMPJump to addressxRETIReturn from interruptxCLCClear ALU carry-inxSETCSet ALU signxSETSSet ALU signxSETSEnable ALU signxCLNClear ALU negativexSETNSet ALU negativexSETNEnable ALU negativexRJUMPRelative jumpx | | | | | |
| SETI Set immediate from register BEQ Branch on equal x BGT Branch on greater than x BGE Branch on greater equal than x BZ Branch on zero x CALL Call function, put return to stack RET Return from function x JUMP Jump to address RETI Return from interrupt CLC Clear ALU carry-in SETC Set ALU carry-in CLS Clear ALU sign SETS Set ALU sign SETS Set ALU sign SETS Set ALU sign SETS Set ALU negative SETN Set ALU negative SETN Set ALU negative RJUMP Relative jump Relative jump SETS Relative jump Relative jump | | - | | | |
| BEQ Branch on equal x BGT Branch on greater than x BGE Branch on greater equal than x BZ Branch on zero x CALL Call function, put return to stack x RET Return from function x JUMP Jump to address x RETI Return from interrupt CLC Clear ALU carry-in SETC Set ALU carry-in CLS Clear ALU sign SETS Set ALU sign SETS Set ALU sign SSETS Enable ALU sign CLN Clear ALU negative SETN Set ALU negative SSETN Enable ALU negative RJUMP Relative jump | | | X | | |
| BGT Branch on greater than x BGE Branch on greater equal than x BZ Branch on zero x **CALL Branch on zero x **CALL Call function, put return to stack x **RET Return from function x **JUMP Jump to address x **RETI Return from interrupt **CLC Clear ALU carry-in **SETC Set ALU carry-in **CLS Clear ALU sign **SETS Set ALU sign **SETS Set ALU sign **SETS Enable ALU sign **CLN Clear ALU negative **SETN Set ALU negative **SETN Enable ALU negative **SETN Enable ALU negative **SETN Enable ALU negative **RETN Relative jump **TEN Set ALU negative **RETN Relative jump **TEN Set ALU negative services and services and services are services and services and services are services are services and services are services | | | N. | | |
| BGE Branch on greater equal than x BZ Branch on zero x **Tegister instructions** CALL Call function, put return to stack x RET Return from function x JUMP Jump to address x RETI Return from interrupt CLC Clear ALU carry-in SETC Set ALU carry-in CLS Clear ALU sign SETS Set ALU sign SETS Set ALU sign SETS Enable ALU sign CLN Clear ALU negative SETN Set ALU negative SETN Fanable ALU negative RJUMP Relative jump | | * | | | |
| BZ Branch on zero x CALL Call function, put return to stack x RET Return from function x JUMP Jump to address x RETI Return from interrupt CLC Clear ALU carry-in SETC Set ALU carry-in CLS Clear ALU sign SETS Set ALU sign SETS Set ALU sign SSETS Enable ALU sign CLN Clear ALU negative SETN Set ALU negative SSETN Return from interrupt CLN Return from interrupt CLS Clear ALU sign SETS Return from interrupt CLS Clear ALU sign SETS Set ALU sign SETS Return from interrupt CLS Clear ALU sign SETS Set ALU sign SETN Return from interrupt SETN Return from function x x x x x x x Enable ALU carry-in CLS Clear ALU sign SETN Set ALU sign SETN Return from function x x x x x x x x Enable ALU negative | 1 | | | | |
| CALL Call function, put return to stack x RET Return from function x JUMP Jump to address x RETI Return from interrupt CLC Clear ALU carry-in SETC Set ALU carry-in CLS Clear ALU sign SETS Set ALU sign SETS Set ALU sign SSETS Enable ALU sign CLN Clear ALU negative SETN Set ALU negative RJUMP Relative jump | 1 | | | | |
| CALL Call function, put return to stack RET Return from function x JUMP Jump to address x RETI Return from interrupt CLC Clear ALU carry-in SETC Set ALU carry-in CLS Clear ALU sign SETS Set ALU sign SSETS Enable ALU sign CLN Clear ALU negative SETN Set ALU negative SSETN Enable ALU negative RJUMP Relative jump | DZ | | A | | |
| RET Return from function x JUMP Jump to address x RETI Return from interrupt CLC Clear ALU carry-in SETC Set ALU carry-in CLS Clear ALU sign SETS Set ALU sign SSETS Enable ALU sign CLN Clear ALU negative SETN Set ALU negative SSETN Enable ALU negative RJUMP Relative jump | CALL | | x | | |
| JUMP Jump to address RETI Return from interrupt CLC Clear ALU carry-in SETC Set ALU carry-in CLS Clear ALU sign SETS Set ALU sign SSETS Enable ALU sign CLN Clear ALU negative SETN Set ALU negative SSETN Enable ALU negative RJUMP Relative jump | | / - | | | |
| RETI Return from interrupt CLC Clear ALU carry-in SETC Set ALU carry-in CLS Clear ALU sign SETS Set ALU sign SSETS Enable ALU sign CLN Clear ALU negative SETN Set ALU negative SSETN Enable ALU negative RJUMP Relative jump | | | | | |
| CLC Clear ALU carry-in SETC Set ALU carry-in CLS Clear ALU sign SETS Set ALU sign SSETS Enable ALU sign CLN Clear ALU negative SETN Set ALU negative SSETN Enable ALU negative RJUMP Relative jump | | <u> </u> | 1 | | |
| SETC Set ALU carry-in CLS Clear ALU sign SETS Set ALU sign SSETS Enable ALU sign CLN Clear ALU negative SETN Set ALU negative SSETN Enable ALU negative RJUMP Relative jump | | | | | |
| CLS Clear ALU sign SETS Set ALU sign SSETS Enable ALU sign CLN Clear ALU negative SETN Set ALU negative SSETN Enable ALU negative RJUMP Relative jump | | | | | |
| SETS Set ALU sign SSETS Enable ALU sign CLN Clear ALU negative SETN Set ALU negative SSETN Enable ALU negative RJUMP Relative jump | | · · | | | |
| SSETS Enable ALU sign CLN Clear ALU negative SETN Set ALU negative SSETN Enable ALU negative RJUMP Relative jump | | | | | |
| CLN Clear ALU negative SETN Set ALU negative SSETN Enable ALU negative RJUMP Relative jump | | | | | |
| SETN Set ALU negative SSETN Enable ALU negative RJUMP Relative jump | | | | | |
| SSETN Enable ALU negative RJUMP Relative jump | 1 | | | | |
| RJUMP Relative jump | | | | | |
| • • | 1 | | | | |
| | | | | | |

Table 2: Instruction set for RISC processor

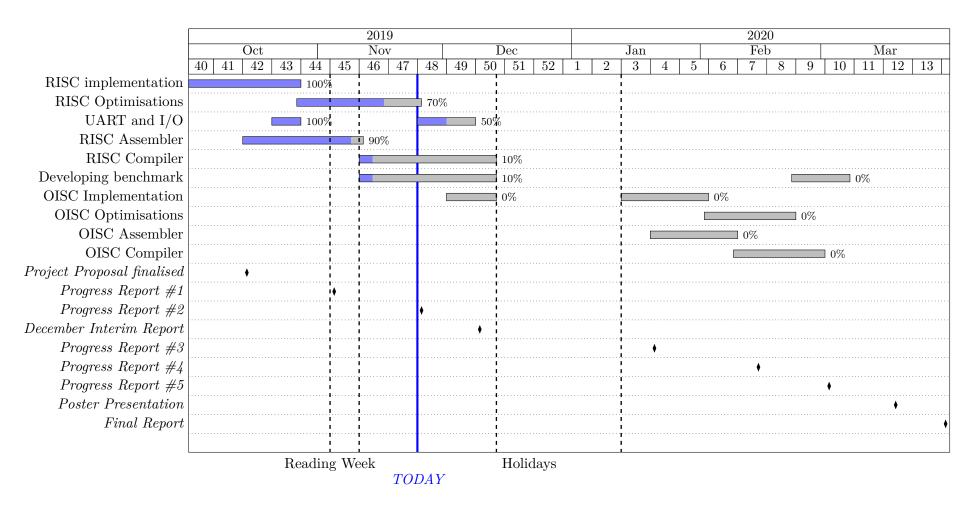


Table 3: Updated project schedule Grantt chart