**Confirmed with Nelson**

- Cannot have scale \* index without a base

- We don't support unaligned access

**Signal Naming Convention:**

*(generating\_unit)*\_*(signal\_name)*\_*(pipeline\_stage)*[\_b if active low]

**de\_SR1\_type\_d2**

mux select

|  |  |
| --- | --- |
| 00 | R/M (from ModR/M) |
| 01 | hardcoded register values |
| 10 | Base |

**de\_SR2\_type\_d2**

mux select

|  |  |
| --- | --- |
| 0 | REG (from ModR/M) |
| 1 | hardcoded register values |

**de\_imm\_type\_d2**

mux select

|  |  |
| --- | --- |
| 0 | imm (from raw instructions) |
| 1 | hardcoded register values |

**cs\_sr1\_d2**

hard coded register id for sr1

**cs\_sr2\_d2**

hard coded register id for sr2

**de\_sib\_exist\_d2**

are we using the sib byte for memory addressing?

|  |  |
| --- | --- |
| 0 | no sib |
| 1 | yes sib |

**de\_opA\_type\_ag**

operand associated with r/m, everything is made to fit. Needs to have possibilities

1) AL/AX/EAX

2) register identified by r/m

3) mem identified by r/m

|  |  |
| --- | --- |
| 00 | reg from SR1 |
| 01 | mem |
| 10 | seg1\_data |
| 11 | seg2\_data |

**de\_opB\_type\_ag**

operand associated with reg, everything is made to fit. Needs to have possibilities

1) register identified by reg

3) immediate

|  |  |
| --- | --- |
| 00 | reg from SR2 |
| 01 | imm |
| 10 | ME\_B |

**de\_seg1\_type\_d2**

selects segment register 1

|  |  |
| --- | --- |
| 0 | D2\_SEGID |
| 1 | 011 (ES) |

**de\_seg2\_type\_d2**

selects segment register 2

|  |  |
| --- | --- |
| 0 | (010) SS |
| 1 | reg (from ModR/M) |

**de\_dest\_type\_ag**

identifies whether second operand is reg from r/m, reg from reg, immediate, or memory value.

|  |  |
| --- | --- |
| 00 | reg from SR1 |
| 01 | reg from SR2 |
| 10 | mem |
|  |  |

**de\_datasize\_all**

size of operand (bits), pseudo-one-hot

|  |  |
| --- | --- |
| 00 | 8 |
| 01 | 16 |
| 10 | 32 |

**de\_addr\_mode\_ag**

each bit stands for the presence of the following to calculate the offset

bit 2: base

bit 1: scale\*index

bit 0: displacement

|  |  |
| --- | --- |
| 001 | displacement |
| 100 | base |
| 101 | base + displacement |
| 110 | base + scale \* index |
| 111 | base + scale \* index + displacement |

**de\_dcache\_read\_me**

does the instruction need to read memory

**de\_operation\_ex**

|  |  |
| --- | --- |
| 000 | ADD |
| 001 | OR |
| 010 | NOT |
| 011 | DAA |
| 100 | AND |
| 101 | CLD |
| 110 | CMP |
| 111 | STD |

**ex\_flags\_affected\_wb\_b**

Which flags are affected. active low, one-hot assignments (only optimization is find out which combinations don't exist in our instruction which is negligible increase in not-the-only-factor cycle time).

|  |  |
| --- | --- |
| 1111110 | CF |
| 1111101 | PF |
| 1111011 | AF |
| 1110111 | ZF |
| 1101111 | SF |
| 1011111 | DF |
| 0111111 | OF |

**Dependency Checking Signals**

**de\_srA\_needed\_ag**

Does the first operand need to read from register?

**de\_srB\_needed\_ag**

Does the second operand need to read from register?

**de\_dr\_needed\_all**

Does the instruction need to write to a register in WB?

**de\_index\_needed\_ag**

Does the addressing mode need to read an index value from register?

**de\_base\_needed\_ag**

Does the addressing need to read a base value from register?

**cs\_seg\_id\_ag**

selects segment

**cs\_operation­\_ex**

operation for ALU