# Instruction Coding A

## A.1 OPCODES

This appendix gives a summary of the complete instruction set of the ADSP-2100 family processors. Opcode field names are defined at the end of the appendix. Any instruction codes not shown are reserved for future use.

Type 1: ALU / MAC with Data & Program Memory Read

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | PD | | DD | | AMF | | | | | Yop | | Xop | | | PM I | | PM M | | DM I | | DM M | |

Type 2: Data Memory Write (Immediate Data)

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | G | DATA | | | | | | | | | | | | | | | | I | | M | |

Type 3: Read /Write Data Memory (Immediate Address)

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | D | RGP | | ADDR | | | | | | | | | | | | | | REG | | | |

Type 4: ALU / MAC with Data Memory Read / Write

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | G | D | Z | AMF | | | | | Yop | | Xop | | | DREG | | | | I | | M | |

Type 5: ALU / MAC with Program Memory Read / Write

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 1 | D | Z | AMF | | | | | Yop | | Xop | | | DREG | | | | I | | M | |

# A Instruction Coding

Type 6: Load Data Register Immediate

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | DATA | | | | | | | | | | | | | | | | DREG | | | |

Type 7: Load Non-Data Register Immediate

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | RGP | | | | DATA | | | | | | | | | | | | REG | | | |

Type 8: ALU / MAC with Internal Data Register Move

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 1 | Z | AMF | | | | | | Yop | | | Xop | | Dest DREG | | | Source DREG | | | |

Generate ALU Status (NONE = <ALU>) *(ADSP-217x, ADSP-218x, ADSP-21msp58/59 only)*

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 1 | 0 | AMF | | | | | | Yop | | | Xop | | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |

ALU codes only

Type 9: Conditional ALU / MAC

xop * yop

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 0 | Z | AMF | | | | | | Yop | | | Xop | | 0 | 0 | 0 | 0 | COND | | |

xop * xop *(ADSP-217x, ADSP-218x, ADSP-21msp58/59 only)*

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 0 | Z | AMF | | | | | | 0 | 0 | | Xop | | 0 | 0 | 0 | 1 | COND | | |

xop AND/OR/XOR constant *(ADSP-217x, ADSP-218x, ADSP-21msp58/59 only)*

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 0 | Z | AMF | | | | | | YY | | | Xop | | CC | | BO | | COND | | |

BO, CC, and YY specify the constant according the table shown at the end of this appendix.

PASS constant (constant ≠ 0,1,–1) *(ADSP-217x, ADSP-218x, ADSP-21msp58/59 only)*

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 0 | Z | AMF | | | | | | YY | | | Xop | | CC | | BO | | COND | | |

# Instruction Coding A

Type 10: Conditional Jump (Immediate Address)

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 | S | ADDR | | | | | | | | | | | | | | COND | | | |

Type 11: Do Until

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 | 1 | ADDR | | | | | | | | | | | | | | TERM | | | |

Type 12: Shift with Data Memory Read / Write

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | G | D | SF | | | | Xop | | | DREG | | | | I | M | | |

Type 13: Shift with Program Memory Read / Write

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | D | SF | | | | Xop | | | DREG | | | | I | M | | |

Type 14: Shift with Internal Data Register Move

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | SF | | | | Xop | | | Dest DREG | | | | Source DREG | | | |

Type 15: Shift Immediate

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | SF | | | | Xop | | | exponent | | | | | | | |

Type 16: Conditional Shift

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | SF | | | | Xop | | | 0 | 0 | 0 | 0 | COND | | | |

# A Instruction Coding

Type 17: Internal Data Move

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | DST RGP | | SRC RGP | | Dest REG | | | | Source REG | | | |

Type 18: Mode Control

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | TI | | MM | | AS | | OL | | BR | | SR | | GM | | 0 | 0 |

Mode Control codes:

| | |
|---|---|
| SR: | Secondary register bank |
| BR: | Bit-reverse mode |
| OL: | ALU overflow latch mode |
| AS: | AR register saturate mode |
| MM: | Alternate Multiplier placement mode |
| GM: | GO Mode; enable means execute internal code if possible |
| TI: | Timer enable |

11 = Enable Mode
10 = Disable Mode
0 1 = no change
0 0 = no change

Type 19: Conditional Jump (Indirect Address)

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | I | | 0 | S | COND | | | |

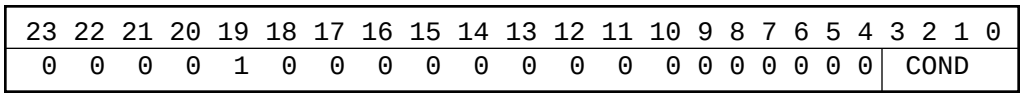
Type 20: Conditional Return

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | T | COND | | | |

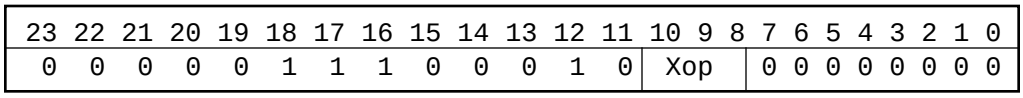
Type 21: Modify Address Register

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | G | I | | M | |

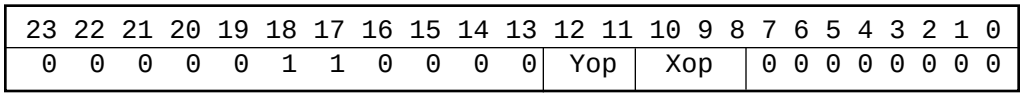
Type 22: Reserved



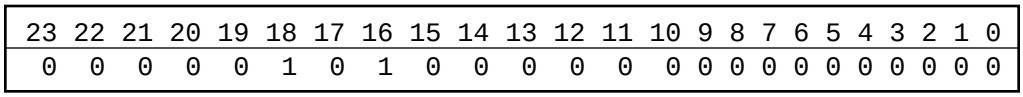

Type 23: DIVQ



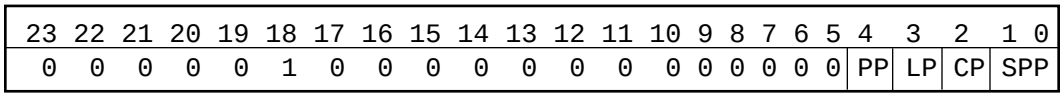

Type 24: DIVS



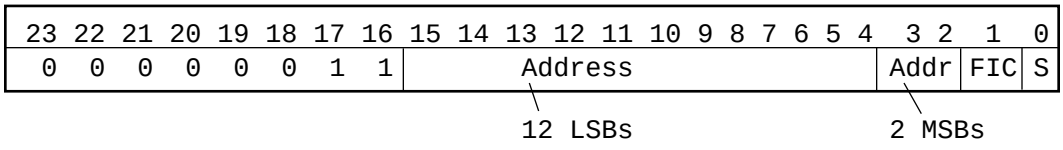

Type 25: Saturate MR



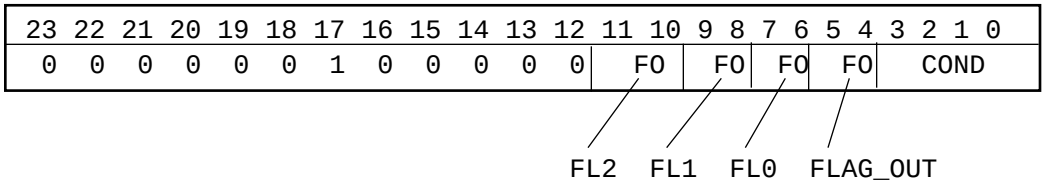

Type 26: Stack Control




Type 27: Call or Jump on Flag In




Type 28: Modify Flag Out



# A Instruction Coding

Type 29: I/O Memory Space Read/Write *(ADSP-218x only)*

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | D | ADDR | | | | | | | | | | | DREG | | | |

Type 30: No Operation (NOP)

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Type 31: Idle

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Type 31: Idle (n) *(Slow Idle)*

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | DV | | | |

## A.2 ABBREVIATION CODING

**AMF** ALU / MAC Function codes

`0 0 0 0 0` No operation

MAC Function codes

| | | | |
|---|---|---|---|
| `0 0 0 0 1` | X * Y | (RND) | |
| `0 0 0 1 0` | MR + X * Y | (RND) | |
| `0 0 0 1 1` | MR – X * Y | (RND) | |
| `0 0 1 0 0` | X * Y | (SS) | Clear when y = 0 |
| `0 0 1 0 1` | X * Y | (SU) | |
| `0 0 1 1 0` | X * Y | (US) | |
| `0 0 1 1 1` | X * Y | (UU) | |
| `0 1 0 0 0` | MR + X * Y | (SS) | |
| `0 1 0 0 1` | MR + X * Y | (SU) | |
| `0 1 0 1 0` | MR + X * Y | (US) | |
| `0 1 0 1 1` | MR + X * Y | (UU) | |
| `0 1 1 0 0` | MR – X * Y | (SS) | |
| `0 1 1 0 1` | MR – X * Y | (SU) | |
| `0 1 1 1 0` | MR – X * Y | (US) | |
| `0 1 1 1 1` | MR – X * Y | (UU) | |

ALU Function codes

| | | |
|---|---|---|
| `1 0 0 0 0` | Y | Clear when y = 0 |
| `1 0 0 0 1` | Y + 1 | PASS 1 when y = 0 |
| `1 0 0 1 0` | X + Y + C | |
| `1 0 0 1 1` | X + Y | X when y = 0 |
| `1 0 1 0 0` | NOT Y | |
| `1 0 1 0 1` | – Y | |
| `1 0 1 1 0` | X – Y + C – 1 | X + C – 1 when y = 0 |
| `1 0 1 1 1` | X – Y | |
| `1 1 0 0 0` | Y – 1 | PASS –1 when y = 0 |
| `1 1 0 0 1` | Y – X | – X when y = 0 |
| `1 1 0 1 0` | Y – X + C – 1 | –X + C – 1 when y = 0 |
| `1 1 0 1 1` | NOT X | |
| `1 1 1 0 0` | X AND Y | |
| `1 1 1 0 1` | X OR Y | |
| `1 1 1 1 0` | X XOR Y | |
| `1 1 1 1 1` | ABS X | |

# A Instruction Coding

**BO** see **YY, CC, BO** at the end of this appendix

**CC** see **YY, CC, BO** at the end of this appendix

**COND** Status Condition codes

| | | |
|---|---|---|
| 0 0 0 0 | Equal | EQ |
| 0 0 0 1 | Not equal | NE |
| 0 0 1 0 | Greater than | GT |
| 0 0 1 1 | Less than or equal | LE |
| 0 1 0 0 | Less than | LT |
| 0 1 0 1 | Greater than or equal | GE |
| 0 1 1 0 | ALU Overflow | AV |
| 0 1 1 1 | NOT ALU Overflow | NOT AV |
| 1 0 0 0 | ALU Carry | AC |
| 1 0 0 1 | Not ALU Carry | NOT AC |
| 1 0 1 0 | X input sign negative | NEG |
| 1 0 1 1 | X input sign positive | POS |
| 1 1 0 0 | MAC Overflow | MV |
| 1 1 0 1 | Not MAC Overflow | NOT MV |
| 1 1 1 0 | Not counter expired | NOT CE |
| 1 1 1 1 | Always true | |

**CP** Counter Stack Pop codes

| | |
|---|---|
| 0 | No change |
| 1 | Pop |

**D** Memory Access Direction codes

| | |
|---|---|
| 0 | Read |
| 1 | Write |

**DD** Double Data Fetch Data Memory Destination codes

| | |
|---|---|
| 0 0 | AX0 |
| 0 1 | AX1 |
| 1 0 | MX0 |
| 1 1 | MX1 |

**DREG** Data Register codes

| | |
|---|---|
| 0 0 0 0 | AX0 |
| 0 0 0 1 | AX1 |
| 0 0 1 0 | MX0 |
| 0 0 1 1 | MX1 |
| 0 1 0 0 | AY0 |
| 0 1 0 1 | AY1 |
| 0 1 1 0 | MY0 |
| 0 1 1 1 | MY1 |
| 1 0 0 0 | SI |
| 1 0 0 1 | SE |
| 1 0 1 0 | AR |
| 1 0 1 1 | MR0 |
| 1 1 0 0 | MR1 |
| 1 1 0 1 | MR2 |
| 1 1 1 0 | SR0 |
| 1 1 1 1 | SR1 |

**DV** Divisor codes for Slow Idle instruction ( *IDLE (n)* )

| | |
|---|---|
| 0 0 0 0 | Normal Idle instruction (Divisor=0) |
| 0 0 0 1 | Divisor=16 |
| 0 0 1 0 | Divisor=32 |
| 0 1 0 0 | Divisor=64 |
| 1 0 0 0 | Divisor=128 |

**FIC** FI condition code

| | | |
|---|---|---|
| 1 | latched FI is 1 | " FLAG_IN " |
| 0 | latched FI is 0 | " NOT FLAG_IN " |

**FO** Control codes for Flag Output Pins (FO, FL0, FL1, FL2)

.

| | |
|---|---|
| 0 0 | No change |
| 0 1 | Toggle |
| 1 0 | Reset |
| 1 1 | Set |

# A Instruction Coding

**G** Data Address Generator codes

| | |
|---|---|
| 0 | DAG1 |
| 1 | DAG2 |

**I** Index Register codes

| G = | 0 | 1 |
|---|---|---|
| 0 0 | I0 | I4 |
| 0 1 | I1 | I5 |
| 1 0 | I2 | I6 |
| 1 1 | I3 | I7 |

**LP** Loop Stack Pop codes

| | |
|---|---|
| 0 | No Change |
| 1 | Pop |

**M** Modify Register codes

| G = | 0 | 1 |
|---|---|---|
| 0 0 | M0 | M4 |
| 0 1 | M1 | M5 |
| 1 0 | M2 | M6 |
| 1 1 | M3 | M7 |

**PD** Dual Data Fetch Program Memory Destination codes

| | |
|---|---|
| 0 0 | AY0 |
| 0 1 | AY1 |
| 1 0 | MY0 |
| 1 1 | MY1 |

**PP** PC Stack Pop codes

| | |
|---|---|
| 0 | No Change |
| 1 | Pop |

**REG** Register codes

Codes not assigned are reserved.

| RGP = | 00 | 01 | 10 | 11 |
|---|---|---|---|---|
| 0 0 0 0 | AX0 | I0 | I4 | ASTAT |
| 0 0 0 1 | AX1 | I1 | I5 | MSTAT |
| 0 0 1 0 | MX0 | I2 | I6 | SSTAT (read only) |
| 0 0 1 1 | MX1 | I3 | I7 | IMASK |
| 0 1 0 0 | AY0 | M0 | M4 | ICNTL |
| 0 1 0 1 | AY1 | M1 | M5 | CNTR |
| 0 1 1 0 | MY0 | M2 | M6 | SB |
| 0 1 1 1 | MY1 | M3 | M7 | PX |
| 1 0 0 0 | SI | L0 | L4 | RX0 |
| 1 0 0 1 | SE | L1 | L5 | TX0 |
| 1 0 1 0 | AR | L2 | L6 | RX1 |
| 1 0 1 1 | MR0 | L3 | L7 | TX1 |
| 1 1 0 0 | MR1 | – | – | IFC (write only) |
| 1 1 0 1 | MR2 | – | – | OWRCNTR (write only) |
| 1 1 1 0 | SR0 | – | – | – |
| 1 1 1 1 | SR1 | – | – | – |

**S** Jump/Call codes

| | |
|---|---|
| 0 | Jump |
| 1 | Call |

# A Instruction Coding

**SF** Shifter Function codes

| | | |
|---|---|---|
| 0 0 0 0 | LSHIFT | (HI) |
| 0 0 0 1 | LSHIFT | (HI, OR) |
| 0 0 1 0 | LSHIFT | (LO) |
| 0 0 1 1 | LSHIFT | (LO, OR) |
| 0 1 0 0 | ASHIFT | (HI) |
| 0 1 0 1 | ASHIFT | (HI, OR) |
| 0 1 1 0 | ASHIFT | (LO) |
| 0 1 1 1 | ASHIFT | (LO, OR) |
| 1 0 0 0 | NORM | (HI) |
| 1 0 0 1 | NORM | (HI, OR) |
| 1 0 1 0 | NORM | (LO) |
| 1 0 1 1 | NORM | (LO, OR) |
| 1 1 0 0 | EXP | (HI) |
| 1 1 0 1 | EXP | (HIX) |
| 1 1 1 0 | EXP | (LO) |
| 1 1 1 1 | Derive Block Exponent | |

**SPP** Status Stack Push/Pop codes

| | |
|---|---|
| 0 0 | No change |
| 0 1 | No change |
| 1 0 | Push |
| 1 1 | Pop |

**T** Return Type codes

| | |
|---|---|
| 0 | Return from Subroutine |
| 1 | Return from Interrupt |

**TERM** Termination codes for DO UNTIL

| | | |
|---|---|---|
| 0 0 0 0 | Not equal | NE |
| 0 0 0 1 | Equal | EQ |
| 0 0 1 0 | Less than or equal | LE |
| 0 0 1 1 | Greater than | GT |
| 0 1 0 0 | Greater than or equal | GE |
| 0 1 0 1 | Less than | LT |
| 0 1 1 0 | NOT ALU Overflow | NOT AV |
| 0 1 1 1 | ALU Overflow | AV |
| 1 0 0 0 | Not ALU Carry | NOT AC |
| 1 0 0 1 | ALU Carry | AC |
| 1 0 1 0 | X input sign positive | POS |
| 1 0 1 1 | X input sign negative | NEG |
| 1 1 0 0 | Not MAC Overflow | NOT MV |
| 1 1 0 1 | MAC Overflow | MV |
| 1 1 1 0 | Counter expired | CE |
| 1 1 1 1 | Always | FOREVER |

**X** X Operand codes

| | |
|---|---|
| 0 0 0 | X0 (SI for shifter) |
| 0 0 1 | X1 (invalid for shifter) |
| 0 1 0 | AR |
| 0 1 1 | MR0 |
| 1 0 0 | MR1 |
| 1 0 1 | MR2 |
| 1 1 0 | SR0 |
| 1 1 1 | SR1 |

**Y** Y Operand codes

| | |
|---|---|
| 0 0 | Y0 |
| 0 1 | Y1 |
| 1 0 | F (feedback register) |
| 1 1 | zero |

# A Instruction Coding

**YY** see **YY, CC, BO** below

**Z** ALU/MAC Result Register codes

| | |
|---|---|
| `0` | Result register |
| `1` | Feedback register |

**YY, CC, BO** ALU / MAC Constant codes (Type 9)

*(ADSP-217x, ADSP-218x, ADSP-21msp58/59 only)*

| *Constant (hex)* | *YY* | *CC* | *BO* | *Bit #* |
|---|---|---|---|---|
| 0001 | 00 | 00 | 01 | bit 0 |
| 0002 | 00 | 01 | 01 | bit 1 |
| 0004 | 00 | 10 | 01 | bit 2 |
| 0008 | 00 | 11 | 01 | bit 3 |
| 0010 | 01 | 00 | 01 | bit 4 |
| 0020 | 01 | 01 | 01 | bit 5 |
| 0040 | 01 | 10 | 01 | bit 6 |
| 0080 | 01 | 11 | 01 | bit 7 |
| 0100 | 10 | 00 | 01 | bit 8 |
| 0200 | 10 | 01 | 01 | bit 9 |
| 0400 | 10 | 10 | 01 | bit 10 |
| 0800 | 10 | 11 | 01 | bit 11 |
| 1000 | 11 | 00 | 01 | bit 12 |
| 2000 | 11 | 01 | 01 | bit 13 |
| 4000 | 11 | 10 | 01 | bit 14 |
| 8000 | 11 | 11 | 01 | bit 15 |
| FFFE | 00 | 00 | 11 | ! bit 0 |
| FFFD | 00 | 01 | 11 | ! bit 1 |
| FFFB | 00 | 10 | 11 | ! bit 2 |
| FFF7 | 00 | 11 | 11 | ! bit 3 |
| FFEF | 01 | 00 | 11 | ! bit 4 |
| FFDF | 01 | 01 | 11 | ! bit 5 |
| FFBF | 01 | 10 | 11 | ! bit 6 |
| FF7F | 01 | 11 | 11 | ! bit 7 |
| FEFF | 10 | 00 | 11 | ! bit 8 |
| FDFF | 10 | 01 | 11 | ! bit 9 |
| FBFF | 10 | 10 | 11 | ! bit 10 |
| F7FF | 10 | 11 | 11 | ! bit 11 |
| EFFF | 11 | 00 | 11 | ! bit 12 |
| DFFF | 11 | 01 | 11 | ! bit 13 |
| BFFF | 11 | 10 | 11 | ! bit 14 |
| 7FFF | 11 | 11 | 11 | ! bit 15 |