{{

17.0.1\_Example\_WRD\_PASM\_Instruction Syntax

Pr0 = $1D8 Pr1 = $1D9

ROR D,{#}S {WC/WZ/WCZ} Rotate right. Note: bit B0 will rotate back to B31

D = [31:0] of ({D[31:0], D[31:0]} >> S[4:0]). C = last bit shifted out if S[4:0] > 0, else D[0]. \*

SHR D,{#}S {WC/WZ/WCZ} Shift right. Note: bit is lost no rotate back

D = [31:0] of ({32'b0, D[31:0]} >> S[4:0]). C = last bit shifted out if S[4:0] > 0, else D[0]. \*

WRC D Write 0 or 1 to D,

according to C. D = {31'b0, C).

}}

CON

\_clkfreq = 200\_000\_000 ''Debug must be enabled clock must be greater than 10MHZ for Debug

VAR

Byte cogRunning 'cog ID started is returned or -1 if not started

Byte PTRAvalue

PUB main()

PTRAvalue := %00000000\_00000000\_00000000\_00000001

Pr0 := %0000000\_00000000\_00000000\_00000001 'cog 0 register

Pr1 := %1000000\_00000000\_00000000\_00000000 'cog 0 register

debug(ubin(S0\_ROR))

debug(ubin(S1\_ROR))

debug(ubin(S2\_ROR))

debug(ubin(S3\_ROR))

debug(ubin(S4\_ROR))

debug(ubin(S5\_ROR))

cogRunning := COGINIT(COGEXEC\_NEW,@S0\_ROR,PTRAvalue)

debug(udec(cogRunning))

repeat

DAT

ORG 0

S0\_ROR ROR Pr0,Pr1 'Execute and shift number of bits in Pr1 B0-> B31

S1\_ROR \_RET\_ ROR Pr0,Pr1 'Execute <inst> always and return if no branch. If not branc pop stack

S2\_ROR IF\_Z ROR Pr0,Pr1 'Execute <inst> if Z = 1.

S3\_ROR IF\_C ROR Pr0,Pr1 'Execute <inst> if C = 1.

S4\_ROR IF\_NC\_AND\_NZ ROR Pr0,Pr1 'Execute <inst> if C = 0 and Z = 0.

S5\_ROR IF\_NZ\_AND\_NC ROR Pr0,Pr1 'Alias Execute <inst> if C = 0 and Z = 0.

debug(ubin(DatVar00),ubin(DatVar01))

ROR DatVar00,#31

WRC DatVar01 ' Write 0 or 1 to D based on C Flag

debug("ROR shift 31 bits", ubin(DatVar00),ubin(DatVar01))

ROR DatVar00,#1 WC

WRC DatVar01

debug(ubin(DatVar00),ubin(DatVar01))

debug(ubin(DatVar02),ubin(DatVar03))

SHR DatVar02,#31

WRC DatVar03

debug("SHR shift 31 bits",ubin(DatVar02),ubin(DatVar03))

SHR DatVar02,#1 WC

WRC DatVar03

debug(ubin(DatVar02),ubin(DatVar03))

\_Loop NOP

JMP #\_Loop 'remember imediate

DatVar00 long %10000000\_00000000\_00000000\_00000000

DatVar01 long 0

DatVar02 long %10000000\_00000000\_00000000\_00000000

DatVar03 long 0