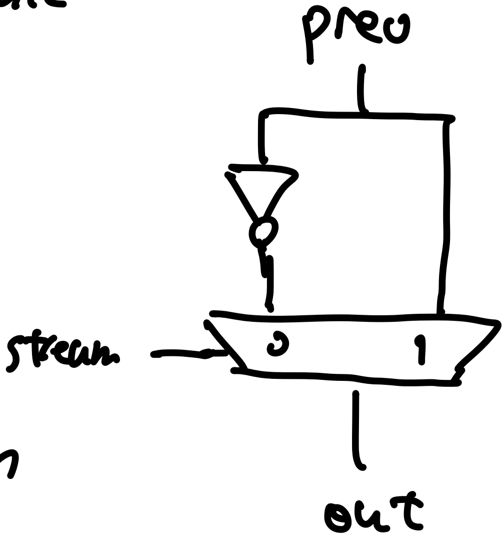
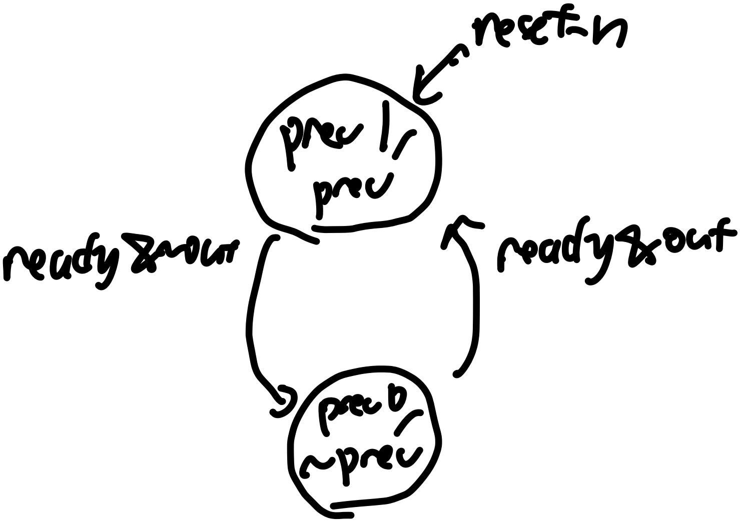
NRZI

ready : Activates the module

stream: The bit stream

out : the output

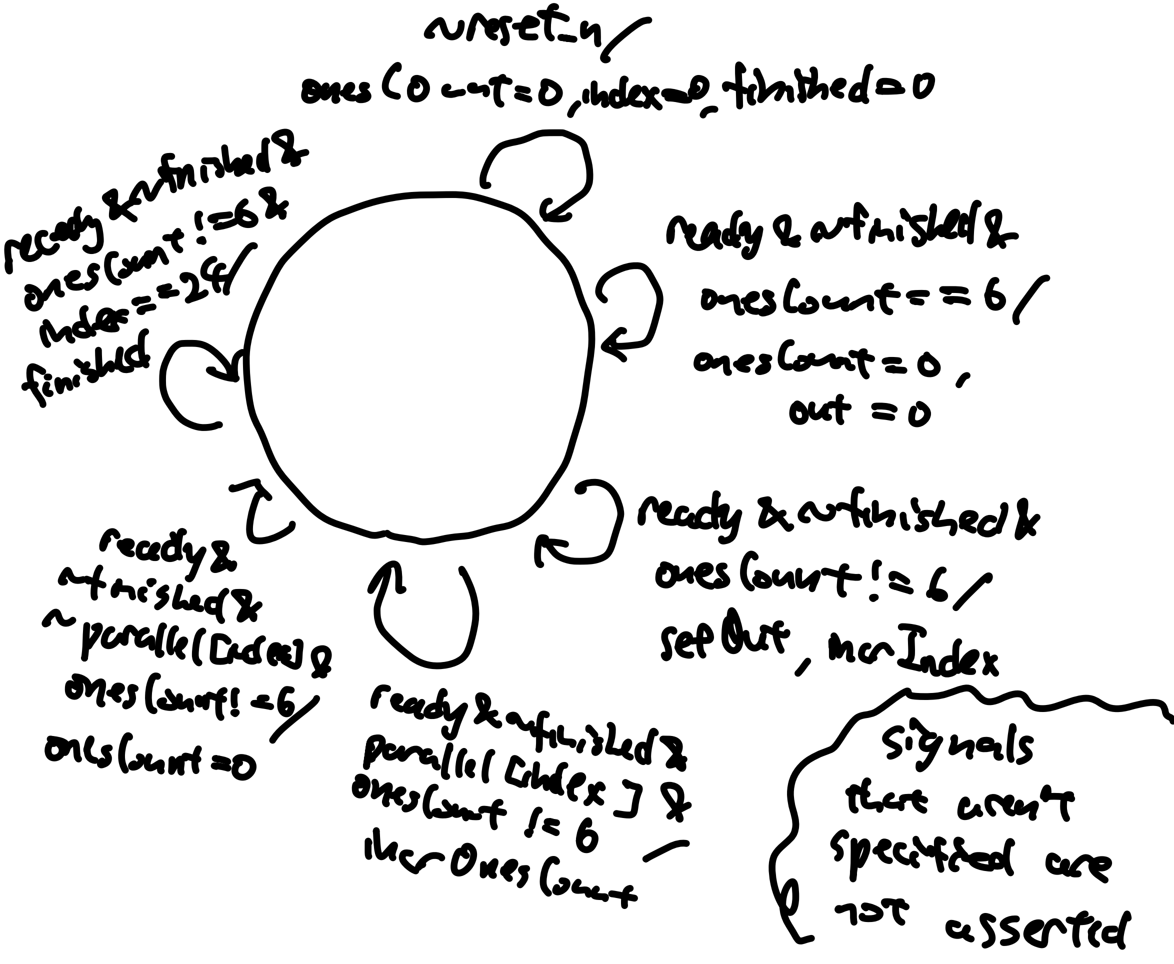




Bit Stuffer

Takes in a parallel 24 bit input, and outputs a bitstream that have been bit stuffed

- ready: Activates module
- parallelIn: The 24-bit input
- out: Output
- finished: Asserted when done processing the input


~reset_n / onesCount=0, index=0, finished=0
ready & ~finished & onesCount != 6 & index == 24 / finished
ready & ~finished & onesCount == 6 / onesCount = 0, out = 0
ready & ~finished & onesCount != 6 / setOut, incrIndex
ready & ~finished & parallel[index] & onesCount != 6 / incrOnesCount
ready & ~finished & ~paralle[index] & onesCount != 6 / onesCount = 0

Signals there aren't specified are not asserted

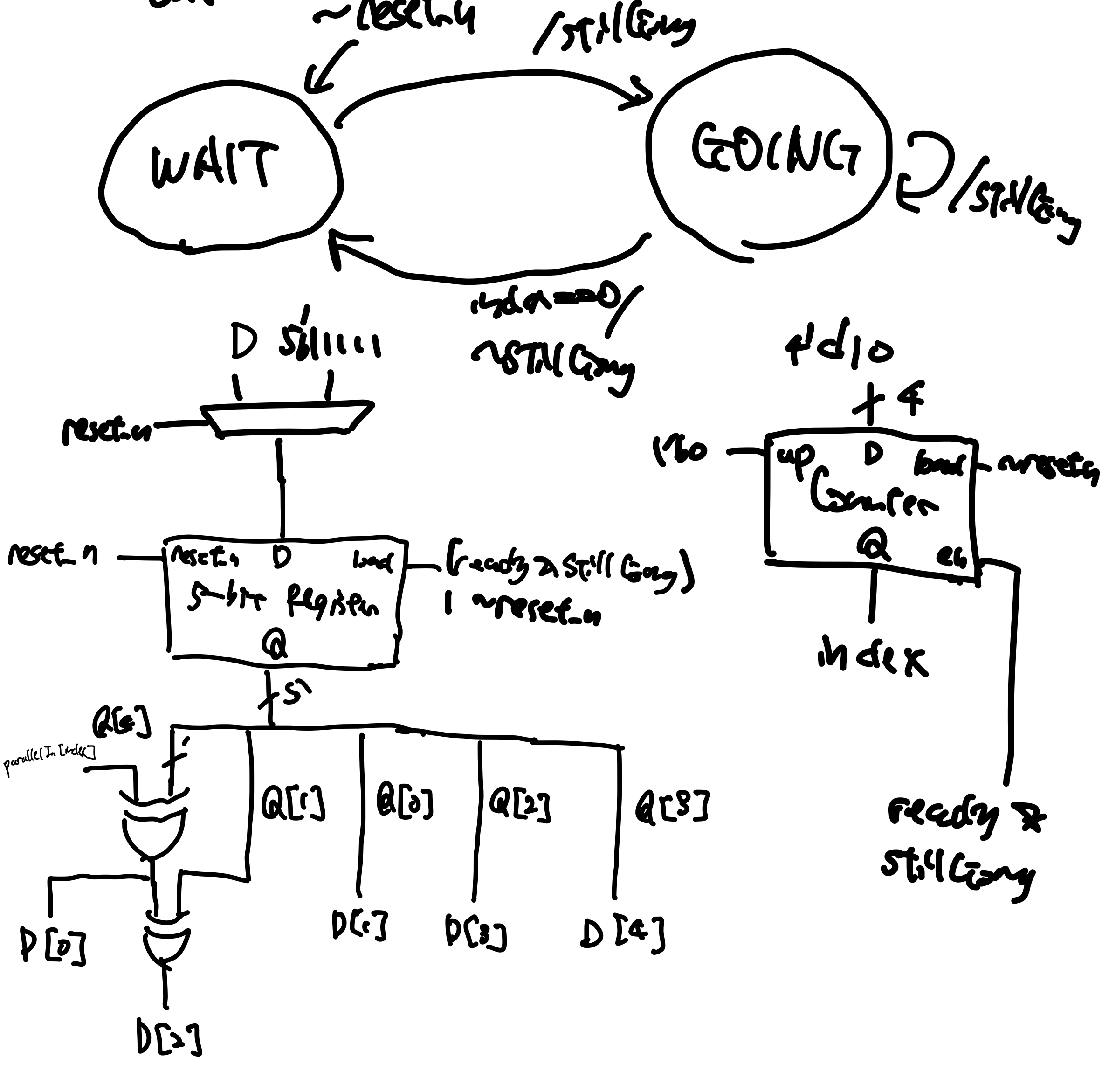
CRC5

ready: Activates module

parallelIn: 11 bit input

done : done calculating CRC5

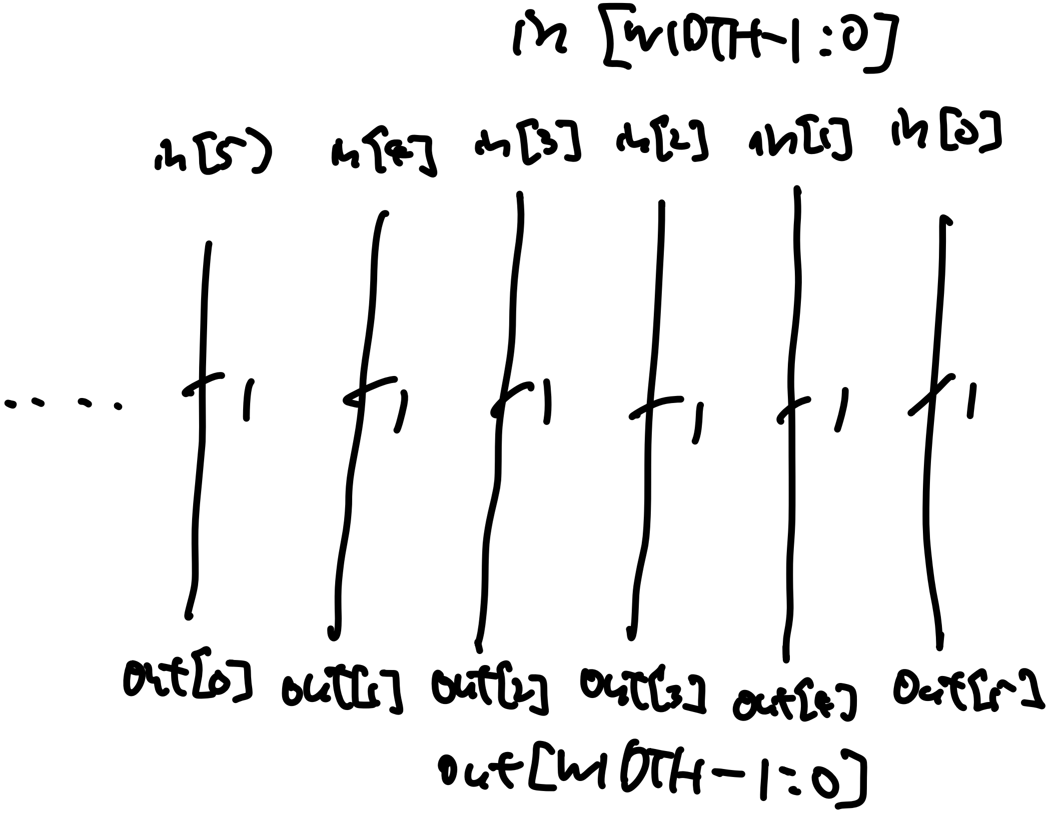
out : 5 bits



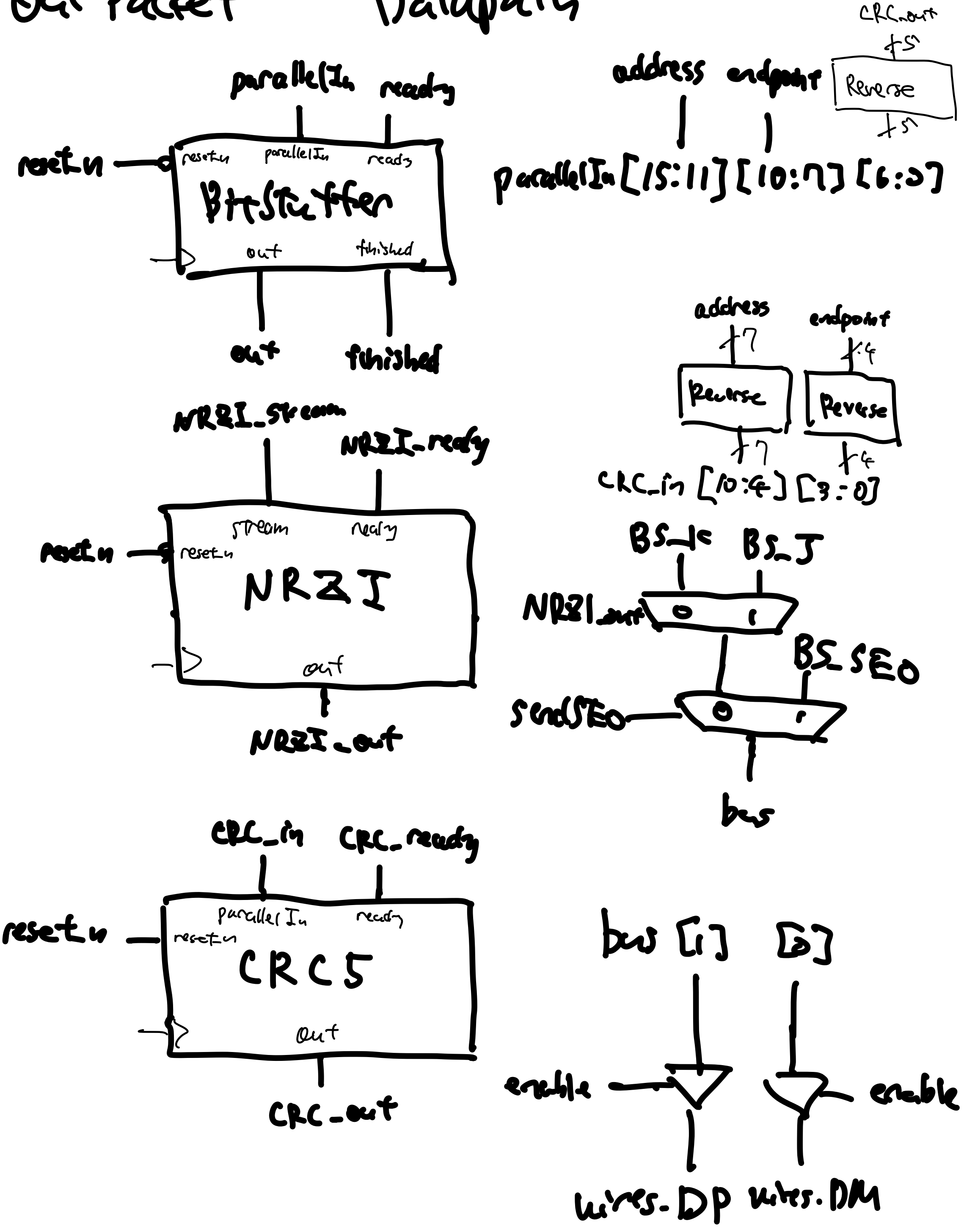
Reverse

Reverses an input.

Usage is changing MSB first to LSB first and vice-versa

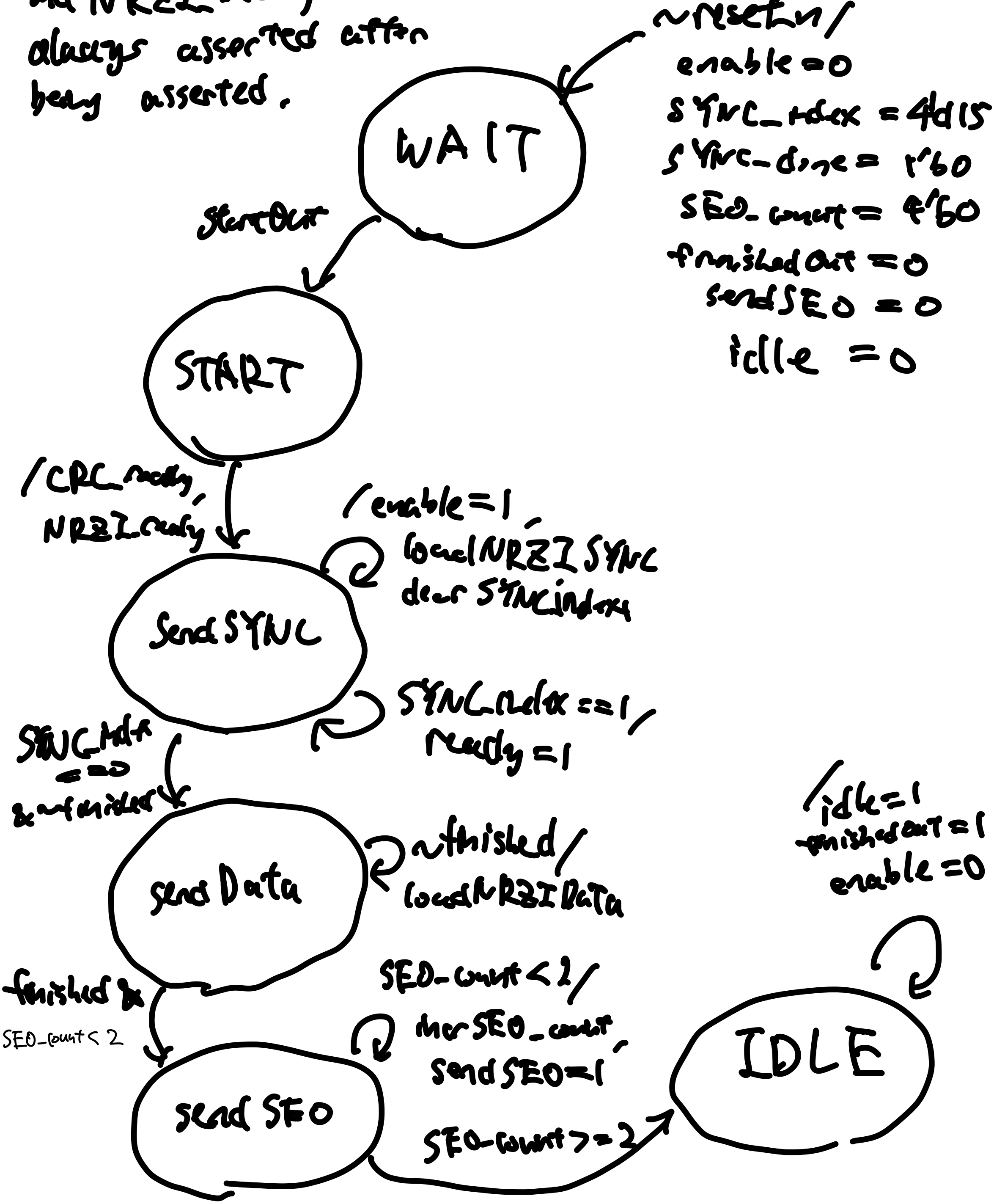


Out Packet Datapath



Assume CRC_ready and NRZI_ready are always asserted after being asserted.

FSM


~reset_n/
enable=0
SYNC_index = 4'd15
SYNC_done = 1'b0
SE0_count = 4'b0
finishedOut = 0
sendSE0 = 0
idle = 0
WAIT
startOut
START
/CRC_ready,
NRZI_ready
/enable=1,
localNRZI SYNC
decr SYNCindex
SendSYNC
SYNCindex ==1/
ready=1
SYNC_index
==0
&~finished
~finished/
localNRZIData
sendData
finished &
SE0_count < 2
SE0_count < 2/
incr SE0_count,
sendSE0=1
send SE0
SE0_count >=2
IDLE
/idle=1
finishedOut=1
enable=0