detect:

clr FineIntervalCount ; Initialise Counters

clr CoarseTimer

detect1:

clr FineTimer

detect2:

cpi CoarseTimer,8 ;If line not idle within 131ms

brlo dl1

rjmp fault ;then exit

dl1:

cpi FineTimer,55 ;If line low for 3.5ms

brge start1 ;then wait for start bit

sbis INPUT\_PORT,INPUT\_PIN

rjmp detect1 // bit is cleared - jump to detect1

rjmp detect2 // bit is set - jump to detect2

start1:

cpi CoarseTimer,8 ;If no start bit detected

brge fault ;within 130ms then exit

sbic INPUT\_PORT,INPUT\_PIN ;Wait for start bit

rjmp start1

// Beginning of first start bit detected

clr FineTimer ;Measure length of start bit

start2:

cpi FineTimer,17 ;If startbit longer than 1.1ms,

brge fault ;exit

sbis INPUT\_PORT,INPUT\_PIN

rjmp start2 ;Positive edge of 1st start bit

mov temp,FineTimer ;timer is 1/2 bit time

clr FineTimer

mov ref1,temp

lsr ref1

mov ref2,ref1

add ref1,temp ;ref1 = 3/4 bit time

lsl temp

add ref2,temp ;ref2 = 5/4 bit time

start3:

cp FineTimer,ref1 ;If high period St2 > 3/4 bit time

brge fault ;exit

sbic INPUT\_PORT,INPUT\_PIN ;Wait for falling edge start bit 2

rjmp start3

clr FineTimer

ldi bitcnt,12 ;Receive 12 bits

clr command

clr system

sample:

cp FineTimer,ref1 ;Sample INPUT at 1/4 bit time

brlo sample

sbic INPUT\_PORT,INPUT\_PIN

rjmp bit\_is\_a\_1 ;Jump if line high

bit\_is\_a\_0: // A data bit '0' has been detected

clc ;Store a '0'

rol command

rol system

;Synchronize timing

bit\_is\_a\_0a:

cp FineTimer,ref2 ;If no edge within 3/4 bit time

brge fault ;exit

sbis INPUT\_PORT,INPUT\_PIN ;Wait for rising edge

rjmp bit\_is\_a\_0a ;in the middle of the bit

clr FineTimer

rjmp nextbit

bit\_is\_a\_1: // A data bit '1' has been detected

sec ;Store a '1'

rol command

rol system

;Synchronize timing

bit\_is\_a\_1a:

cp FineTimer,ref2 ;If no edge within 3/4 bit time

brge fault ;exit

sbic INPUT\_PORT,INPUT\_PIN ;Wait for falling edge

rjmp bit\_is\_a\_1a ;in the middle of the bit

clr FineTimer

nextbit:

dec bitcnt ;If bitcnt > 0

brne sample ;get next bit

;All bits sucessfully received!

mov temp,command ;Place system bits in "system"

rol temp

rol system

rol temp

rol system

bst system,5 ;Move toggle bit

bld command,6 ;to "command"

;Clear remaining bits

andi command,0x7F // Toggle, + 6 'command' bits

andi system,0x1F // 5 'address' bits

//Display "Command" value on PortB (LEDs)

// com command

// out PORTB,command ; out to LEDs

ret

FAULT: ;Set both "command" and "system" to 0xFF to indicate failure

ldi command, 0xFF ;Set all bits in "command"

ldi system, 0xFF ;Set all bits in "system"

ret

// \*\*\*\*\* END of IR data detection code \*\*\*\*\*