#------------------------- THYMIO - BEACON - BROADCAST - RECEIVER OR SENDER ------- ACRONYM (BBRS)

#-------------------- THERE ARE TWO MOST IMPORTANT BEHAVIOUR TO EXPECT FROM THE BBRS ---------

#----------------- 1. THYMIO HAS BROADCAST SENDER

#---------------------- FOR BBRS TO SEND MESSAGE FROM A KNOW BBRS\_ID IT MUST PASSED THE MESSAGE TO A TO DELIVERY SERVICE ROBOT

#------------------ 2. THYMIO HAS BROADCAST RECEIVER

#----------------------------- FOR BBRS TO RECEIVE TO MESSAGE ITS ID MUST MATCH THE CORRESPONDING DESTINATION\_ID BROADCASTED BY THE SERVICE DELIVERY

#------------------ RESET SUBROUTINES -------------------------

# A reset is necessary when robots get wrong value

# I have defined a threshold to improve performance of message passed

# 1. ID TO BE ASSIGNED MUST BE DIVISIBLE BY PRIME NUMBERS 3 AND 5 eg 15, 45

#i,e, 15,30,45,60,75,90,105,120,135,150,165,180,195,210,225,240,255,270,285,300

# 2. THE ACKS TO BE SENT MUST BE THE SENDER\_ID + 1000 AND DESTINATION\_ID + 1001 AND MESSAGE + 1

# THRESHOLD FOR SENDER\_ID ACK and DESTINATION\_ID ACK IS BETWEEN (1000 - 1056)

# THRESHOLD FOR MESSAGE IS (MESSAGE + 1)

# THIS IS CALLED WHEN WRONG VALUES ARE PASSED TO RESET ALL VARIABLES TO DEFAULTS var init\_RESET = 0

var init\_BBRS\_as\_Sender = 1# --set BBRS AS SENDER

var init\_BBRS\_as\_Reciever = 0 # --set BBRS AS RECEIVER

#-------------default variables--

var BBRS\_ID = 15

var SENDER\_ID = BBRS\_ID

var DESTINATION\_ID = 30 # THIS VARIABLE IS ASSOCIATED TO A KNOWN BBRS WITH ID 55

var MESSAGE = 20 # message to sent using ASCII CODE [ ~ hi ~ ] without the zero

#----------required ACKS FROM SERVICE DELIVERY to ensure message has being passed

var ACK\_SENDER\_ID = 0 # THIS IS RECEIVED FROM SERVICE DELIVERY

var ACK\_DESTINATION\_ID = 0 # THIS IS RECEIVED FROM SERVICE DELIVERY

var ACK\_MESSAGE = 0

#------------------------------------------- 3 ACKS REQUIRED TO VERIFY THE VALUE PASSED BETWEEN BBRS AND DS

var ACK\_SENDER\_ID\_RCV = 0

var ACK\_DESTINATION\_ID\_RCV = 0

var ACK\_MESSAGE\_ID\_RCV = 0

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_VARIABLE COUNTER FOR TIMERS

var countSetTimer0 = 0

var timerWait = 0

var initTimer2ForSound = 0

#Delivery Details is stored here

var MESSAGE\_FROM\_SENDER = 0

var DS\_FOUND = 0

var NEUTRAL\_ACKS\_SENT = 0 # THIS ACK IS SENT BY RECEIVER

var COUNT\_TO\_RESET\_FOR\_SENDER = 0

var COUNT\_TO\_RESET\_FOR\_RECEIVER = 0

var INIT\_RESET = 0 # this triggered only when BBRS reciever has received message passed through the chain

call prox.comm.enable(1)

call leds.top(32,32,32)

call sound.system(5)

#\_+\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_PROX EVENT TO SEND AND RECIEVE MESSAGE

#HERE THE SENDER INITIATE COMMUNICATION BY SENDING OWN BBRS\_ID

if (init\_BBRS\_as\_Sender == 1 and ACK\_SENDER\_ID\_RCV == 0 and ACK\_DESTINATION\_ID\_RCV ==0 and ACK\_MESSAGE\_ID\_RCV ==0 and INIT\_RESET == 0) then

timer.period[0] = 200

prox.comm.tx = SENDER\_ID#---------------BROADCAST OWN BBRS\_ID

end

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_TIMER 0

onevent timer0

if (init\_BBRS\_as\_Sender ==1) then

#call sound.system(5)

countSetTimer0++

call leds.top(32,0,0)

if (countSetTimer0 > 0) then

call leds.top(0,0,0)

countSetTimer0 = 0

timer.period[0] = 0 # timer sleep

initTimer2ForSound =1 # variable required so that timer 1 is reusable

timer.period[1] = 500 # init timer 1

end

end

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_TIMER 1

onevent timer1

if (init\_BBRS\_as\_Sender == 1) then

if (initTimer2ForSound == 1) then

timerWait++

if (timerWait > 2) then

call leds.top(32,0,0)

# call sound.system(5)

timer.period[0] = 200

timerWait = 0

end

end

end

##### RESET AFTER COUNTDOWN ELAPSE FOR THE RECEIVER

if(NEUTRAL\_ACKS\_SENT == 1 and init\_BBRS\_as\_Reciever == 1 ) then

prox.comm.tx = 202 # this message is passed back to indicate that message received and the chain turn green

COUNT\_TO\_RESET\_FOR\_RECEIVER++

if (COUNT\_TO\_RESET\_FOR\_RECEIVER > 70) then

prox.comm.tx = -1

DS\_FOUND = 0

MESSAGE\_FROM\_SENDER = 0

MESSAGE = 0

NEUTRAL\_ACKS\_SENT = 0

call leds.top(0,0,32) # RESET COMPLETE

COUNT\_TO\_RESET\_FOR\_RECEIVER = 0

timer.period[1] = 0

end

end

### RESET AFTER COUNTDOWN ELAPSE FOR SENDER

# AS SENDER

if (INIT\_RESET == 1 and init\_BBRS\_as\_Sender ==0) then

call leds.top(0,32,0)

COUNT\_TO\_RESET\_FOR\_SENDER++

if (COUNT\_TO\_RESET\_FOR\_SENDER > 100) then

#reset all ack recieved

ACK\_SENDER\_ID = 0

ACK\_SENDER\_ID\_RCV = 0

ACK\_DESTINATION\_ID = 0

ACK\_DESTINATION\_ID\_RCV = 0

ACK\_MESSAGE = 0

ACK\_MESSAGE\_ID\_RCV = 0

INIT\_RESET = 0

##broadcast ID again

prox.comm.tx = SENDER\_ID

COUNT\_TO\_RESET\_FOR\_SENDER = 0

init\_BBRS\_as\_Sender = 1

call prox.comm.enable(1)

timer.period[1] = 0

end

end

onevent prox.comm

## THE SENDER WILL EXPECT A REPLY PASSED THROUGH THE CHAIN TO TURN GREEN THEN INITS TIMER TO RESET

if(prox.comm.rx == 202 and init\_BBRS\_as\_Reciever ==0)then

call leds.top(0,32,0)

# INIT\_RESET = 1

call prox.comm.enable(0)

timer.period[1] = 200

end

#CODE FOR BBRS HAS SENDER STARTS HERE###############################################################

if (init\_BBRS\_as\_Sender == 1 and INIT\_RESET !=1) then #------------------------------------ BEHAVIOUR HAS A BROADCAST SENDER

if (ACK\_SENDER\_ID < 1 and prox.comm.tx == SENDER\_ID) then

callsub GET\_ACK\_SENDER\_ID # an ack expected from DS

end

if (ACK\_SENDER\_ID\_RCV == 1 and ACK\_DESTINATION\_ID < 1) then

call leds.top(32,5,0)

prox.comm.tx = DESTINATION\_ID # SEND DESTINATION\_ID SINCE WE NOW HAVE CONFIRMATION FOR SENDER\_ID SENT

callsub GET\_ACK\_DESTINATION # call subroutine to get the ack for destination\_id sent

end

#if conditions above are met i.e when we have sent required ids and recieved ack for sender\_id and destination\_id sent

#now we can send the main message to complete the 3 - way handshake

if (ACK\_DESTINATION\_ID\_RCV == 1 and ACK\_MESSAGE < 1) then

call leds.top(32,10,0)

prox.comm.tx = MESSAGE # send the main message

callsub GET\_ACK\_MESSAGE # get the ack for message sent

end

end

# now that we have recieved all ACK for sender\_id , destination\_id and message sent to DS from BBRS

if(ACK\_SENDER\_ID\_RCV == 1 and ACK\_DESTINATION\_ID\_RCV ==1 and ACK\_MESSAGE\_ID\_RCV ==1 and INIT\_RESET !=1) then

call leds.top(0,32,0) # turn green

##prox.comm.tx =-1

init\_BBRS\_as\_Sender = 0

prox.comm.tx = -1

end

# CODE FOR BBRS AS SENDER ENDS END HERE |----> ###########################################

############################################################################

############################################################################

############################################################################

############################################################################

############################################################################

# CODE SECTION FOR RECEIVER

# CODE FOR BBRS RECEIVER STARTS HERE##################################################

#listen for broadcast

#assign default broadcast value is 445 indicates that DS is in range with data from sender

if (init\_BBRS\_as\_Reciever == 1) then

if (prox.comm.rx == 445 and MESSAGE\_FROM\_SENDER < 1) then

DS\_FOUND = 1

end

if (DS\_FOUND == 1 and MESSAGE\_FROM\_SENDER <1) then

#send your BBRS\_ID

prox.comm.tx = BBRS\_ID

#GET THE MESSAGE FROM DS

if (prox.comm.rx != BBRS\_ID and prox.comm.rx != 446 and prox.comm.rx !=445 and MESSAGE\_FROM\_SENDER < 1) then

MESSAGE\_FROM\_SENDER = prox.comm.rx # save the message |/|/|/|/|/|/|

#send ack to DS\_SENDER

prox.comm.tx = MESSAGE\_FROM\_SENDER + 1

# NEUTRAL\_ACKS\_SENT = 1

call leds.top(0,32,0)

end

end

end

# MESSAGE FROM SENDER WOULD HAVE BEEN RECEIVED AT THIS POINT

if (NEUTRAL\_ACKS\_SENT == 1) then

call leds.top(0,32,0) ##receuved indicator

call sound.system(5)

#NEUTRAL\_ACKS\_SENT = 0

## when all message passed successfully init timer

timer.period[1] = 200 # timer to trigger reset

end

## CODE SEGMENT FOR RECEIVER ENDS HERE#################################################

#SUBROUTINES FOR SENDING REQUIRED VALUE TO DS

sub GET\_ACK\_SENDER\_ID

#this happens only if the ACK\_SENDER\_ID = 0 and prox.comm.tx = SENDER\_ID at the moment

if (ACK\_SENDER\_ID < 1 and prox.comm.tx == SENDER\_ID) then

#now get the value of prox.comm.rx if its passed the simple computation

#i.e we verify what the ACK\_SENDER\_ID sent from the DS

if (prox.comm.rx == (SENDER\_ID + 1000)) then

ACK\_SENDER\_ID =prox.comm.rx #save the ACK recieved

ACK\_SENDER\_ID\_RCV = 1

end

end

sub GET\_ACK\_DESTINATION

#this happens only if we have recvd confirmation that ack for the SENDER\_ID sent

if (ACK\_DESTINATION\_ID <1 and ACK\_SENDER\_ID\_RCV == 1) then

#just to be sure we get the right value we assert the condition again

#now we verify the value of the prox.comm.rx

if (prox.comm.rx !=ACK\_SENDER\_ID and prox.comm.rx == (1001+DESTINATION\_ID)) then # we require value not the ack recvd for SENDER\_ID and if meet the ack computation

ACK\_DESTINATION\_ID = prox.comm.rx # save the ACK recieved

ACK\_DESTINATION\_ID\_RCV = 1

end

end

sub GET\_ACK\_MESSAGE

#this only if we have recieved both ack for sender\_id and destination sent

if (ACK\_MESSAGE <1 and ACK\_DESTINATION\_ID\_RCV ==1) then

#now we verify the value of prox.comm.enable

if (prox.comm.rx != ACK\_DESTINATION\_ID and prox.comm.rx == (MESSAGE + 1)) then

ACK\_MESSAGE = prox.comm.rx

ACK\_MESSAGE\_ID\_RCV = 1

end

end