1. Construct a Raspberry Pi Model Robot System for child monitoring. The robot should take care of the child in absence of parents. It should entertain the child and also alert the parents in case of any emergency.

Identify the tasks and subtasks in the above process (Critical / non-critical / exceptional). Develop the python script for the same. No need to do the circuit simulation. Draw a block diagram of the model illustrating the components that would be used.

Identify the scheduling algorithm which would be effective in tasks modelling and implement the same using Cheddar. (No need to develop hardware model). Writeup should have the following

* Python Script
* Block diagram of model
* Algorithm to be used for scheduling
* Cheddar screen shot

**Answer :**

**Python Script**

**ChildMonitoringRobot.py**

import SelfTest

import random

from datetime import datetime

from time import sleep

from SelfTest import PerformTest

print "[" + str(datetime.now()) + "]: " + "Child Monitoring robot starting up..."

sleep(0.5)

testResult = PerformTest()

print ""

if(testResult == 0):

print "[" + str(datetime.now()) + "]: " + "Child Monitoring robot boot successfull"

onMode = True

naptime = datetime.now()

naptime = naptime.replace(hour = 21, minute = 0, second = 0, microsecond = 0)

while (onMode):

parentStatus = raw\_input("Are parents home? (Y/N)")

if (parentStatus == "N" or parentStatus == "n"):

emergencyNumber = raw\_input("Enter number in case of an emergency")

workingMode = True

while (workingMode):

print ""

print "Monitor mode: 0"

print "Play mode: 1"

print "Feed mode: 2"

print "Clean mode: 3"

print "Sleep mode: 4"

modeSelect = raw\_input("Select a mode")

print ""

if (datetime.now() >= naptime and modeSelect != "4"):

print "[" + str(datetime.now()) + "]: " + "Overriding mode"

modeSelect = "4"

if modeSelect == "0":

print "[" + str(datetime.now()) + "]: " + "Switchting to monitor mode..."

print "[" + str(datetime.now()) + "]: " + "Monitoring baby's movements and ensuring safety."

elif modeSelect == "1":

print "[" + str(datetime.now()) + "]: " + "Switchting to play mode..."

print "[" + str(datetime.now()) + "]: " + "Fetching baby's toys to the play area"

sleep(0.5)

print "[" + str(datetime.now()) + "]: " + "Moving baby to the play area"

sleep(0.5)

print "[" + str(datetime.now()) + "]: " + "Monitoring baby's movements and ensuring safety."

elif modeSelect == "2":

print "[" + str(datetime.now()) + "]: " + "Switchting to feed mode..."

print "[" + str(datetime.now()) + "]: " + "Preparing baby's food"

sleep(0.5)

print "[" + str(datetime.now()) + "]: " + "Moving baby to the feeding chair"

sleep(0.5)

print "[" + str(datetime.now()) + "]: " + "Feeding the baby"

elif modeSelect == "3":

print "[" + str(datetime.now()) + "]: " + "Switchting to clean mode..."

print "[" + str(datetime.now()) + "]: " + "Fetching new diapers"

sleep(0.5)

print "[" + str(datetime.now()) + "]: " + "Moving baby to the washroom"

sleep(0.5)

print "[" + str(datetime.now()) + "]: " + "Changing the baby into new diapers"

elif modeSelect == "4":

print "[" + str(datetime.now()) + "]: " + "Switchting to bedtime mode..."

print "[" + str(datetime.now()) + "]: " + "Preparing baby's bed/crib"

sleep(0.5)

print "[" + str(datetime.now()) + "]: " + "Moving baby to the bed/crib"

sleep(0.5)

print "[" + str(datetime.now()) + "]: " + "Rocking the crib at 5% capacity"

else:

print "[" + str(datetime.now()) + "]: " + "Switchting to emergency mode..."

print "[" + str(datetime.now()) + "]: " + "Contacting parents on number: " + emergencyNumber

workingMode = False

sleep(5)

else:

sleep(2)

print ""

tempResponse = raw\_input("Remain switched on? (Y/N)")

if (tempResponse == "N" or tempResponse == "n"):

onMode = False

print "[" + str(datetime.now()) + "]: " + "Child Monitoring robot Shutting down"

else:

onMode = True

else:

print "[" + str(datetime.now()) + "]: " + "Child Monitoring robot Shutting down"

**SelfTest.py**

import random

from datetime import datetime

from time import sleep

from Maintenance import PerformMaintenance

def PerformTest():

task = random.randint(0,1)

print ""

print "[" + str(datetime.now()) + "]: " + "Performing Self test..."

sleep(0.5)

if (task == 1) :

print "[" + str(datetime.now()) + "]: " + "Self test found some issues"

task = PerformMaintenance()

else :

print "[" + str(datetime.now()) + "]: " + "Self test successful"

return task;

**Maintenance.py**

import random

from datetime import datetime

from time import sleep

def PerformMaintenance():

print ""

print "[" + str(datetime.now()) + "]: " + "Running Maintenance..."

sleep(0.5)

task = random.randint(0,1)

if (task == 1):

errors = random.randint(1,5)

print "[" + str(datetime.now()) + "]: " + str(errors) + " issues were unresolved. Kindly contact the service provider with the following error codes:"

i = 0

while(i < errors):

print "[" + str(datetime.now()) + "]: " + "ER0X" + str(random.randint(1123123, 123874593))

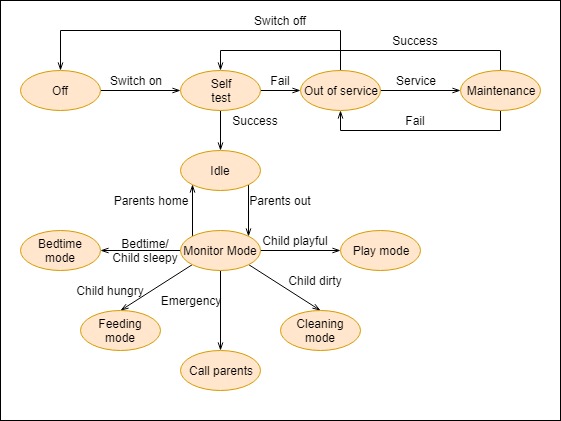
i = i + 1

else:

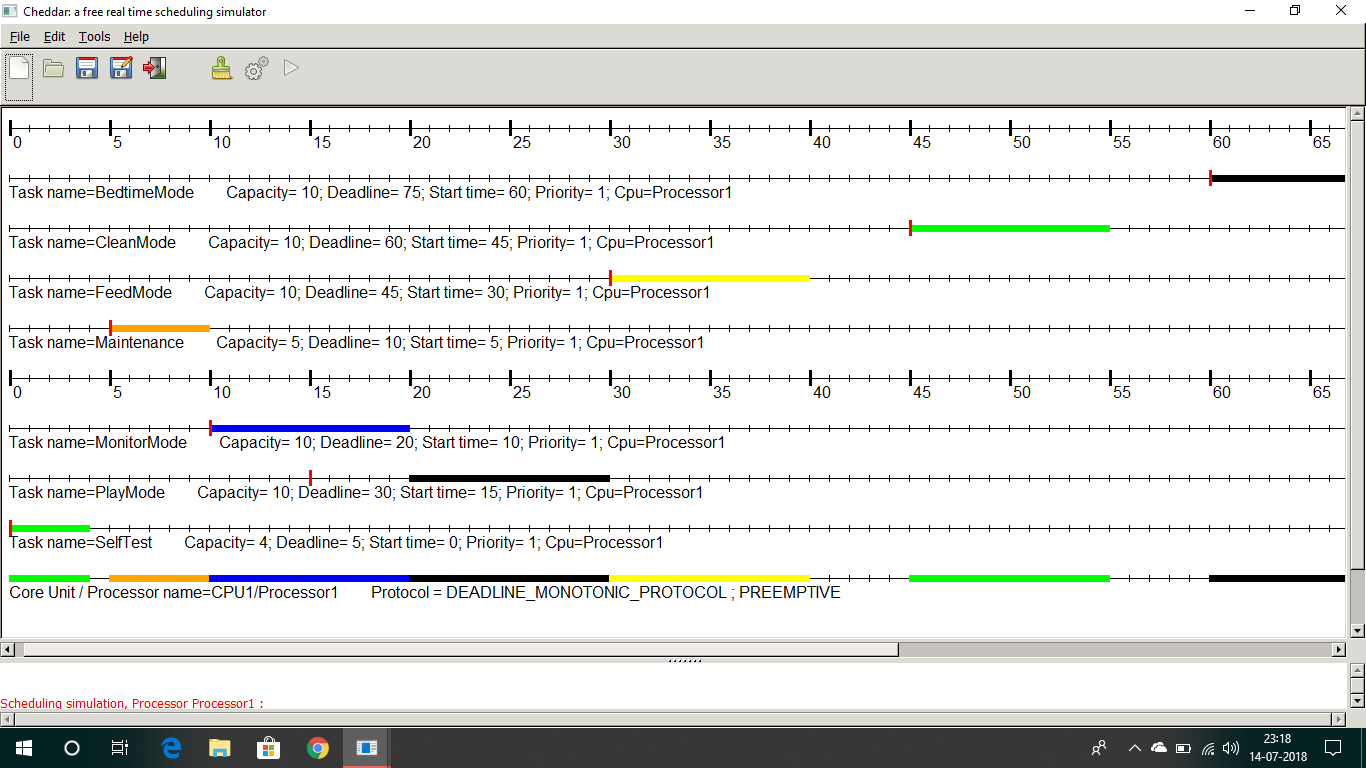
print "[" + str(datetime.now()) + "]: " + str(random.randint(1,5)) + " issues found and rectified."

return task;

Block diagram of model



Algorithm to be used for scheduling



Cheddar screen shot

