The Software Build and Test

I want this page to contain the Software for the Clock, and for some test patterns, for now here is the code for the clock (remember, this code is for Common Anode displays)

This code originated from the following website (although I am pretty sure that is not the original source)

http://raspi.tv/2015/how-to-drive-a-7-segment-display-directly-on-raspberry-pi-in-python

This code was then modified by myself and a couple of friends (thanks to Malc and Chewie) to account for a couple of changes, and these are:

1. My 7 Segment LED is Common Anode, the original code is for common cathode.
2. My 7 Segment LED has a time separator and full stops (so is 8 segments) the original does not.
3. I used different GPIO lines.

Working Clock Code

This is the final version of code that was used. It has the support for the flashing dots time separator. If cutting and pasting isn't working well for you, download the python file

Here

.

```
#! /usr/bin/python
# Python Script to run a 7 Segment Common Anode LED as a clock.
# Version 1.0
import RPi.GPIO as GPIO
import time

#Define numbering system for the IO pins Raspberry Pi
GPIO.setmode(GPIO.BCM)

# Define GPIO ports for the 7seg
gpioSegments =  (5,6,13,19,26,21,20,16)
# 7seg_segment_pins (11,7,4,2,1,10,5,3) +  100R inline
# Setup channels for output and set initial values
for segment in (gpioSegments):
  GPIO.setup(segment,GPIO.OUT)
  GPIO.output(segment,GPIO.HIGH)
# Define GPIO ports for the digits 0-3
gpioDigits = (18,23,24,25)
# Pins (12,9,8,6) select digits 0-3 respectively reading LTR on display
# Setup channels for output and set initial values
for digit in (gpioDigits):
  GPIO.setup(digit,GPIO.OUT)
  GPIO.output(digit,GPIO.LOW)

#Define flags for comtrol of Colon separator on display
colon_visible=True
colon_counter=0

#Define segment arrays for each number to be displayed
numbers = {
    '0':(0,0,1,0,1,0,0,0),
    '1':(1,1,1,0,1,0,1,1),
    '2':(0,0,1,1,0,0,1,0),
    '3':(1,0,1,0,0,0,1,0),
    '4':(1,1,1,0,0,0,0,1),
    '5':(1,0,1,0,0,1,0,0),
    '6':(0,0,1,0,0,1,0,0),
```

```
    '7':(1,1,1,0,1,0,1,0),
    '8':(0,0,1,0,0,0,0,0),
    '9':(1,0,1,0,0,0,0,0),
    ' ':(0,0,0,0,0,0,0,0)}

#Cycle through each digit and its segments
try:
    while True:
        time_string = str(time.ctime()[11:13]+time.ctime()[14:16]).rjust(4)
        for digit in range(4):
#select digit to display
            GPIO.output(gpioDigits[digit], 1)
#set required segments on
            for segment in range(0,8):
                GPIO.output(gpioSegments[segment], numbers[time_string[digit]][segment])
#check to see if we are on segment 3 of digit 2(LTR)
                if ((digit==1) and (segment==2)):
#when colon counter gets to set value flip colon display mode between TRUE (visible) and False
(NOT.visible)
                    if (colon_counter<=50):
#count value not reached turn colon ON or OFF based on current setting of colon_visible
                        if colon_visible==True:
                            GPIO.output(13, 0)
                        else:
                            GPIO.output(13, 1)
#colon counter set value reached so flip colon display mode and reset colon counter
                    else:
                        colon_counter = 0
                        colon_visible =  not colon_visible
#display all selected segments for a short time
            time.sleep(0.005)
#turn-off All segments
            for segment in range(0,8):
                GPIO.output(gpioSegments[segment],GPIO.HIGH)
#turn-off current digit selector pin
            GPIO.output(gpioDigits[digit], GPIO.LOW)
#advance colon counter
        colon_counter=colon_counter+1
finally:
  GPIO.cleanup()
```

Here is the original code, unedited

```
# code modified, tweaked and tailored from code by bertwert
# on RPi forum thread topic 91796
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)

# GPIO ports for the 7seg pins
segments =  (11,4,23,8,7,10,18,25)
# 7seg_segment_pins (11,7,4,2,1,10,5,3) +  100R inline

for segment in segments:
    GPIO.setup(segment, GPIO.OUT)
    GPIO.output(segment, 0)

# GPIO ports for the digit 0-3 pins
digits = (22,27,17,24)
# 7seg_digit_pins (12,9,8,6) digits 0-3 respectively

for digit in digits:
```

```
    GPIO.setup(digit, GPIO.OUT)
    GPIO.output(digit, 1)

num = {' ':(0,0,0,0,0,0,0),
    '0':(1,1,1,1,1,1,0),
    '1':(0,1,1,0,0,0,0),
    '2':(1,1,0,1,1,0,1),
    '3':(1,1,1,1,0,0,1),
    '4':(0,1,1,0,0,1,1),
    '5':(1,0,1,1,0,1,1),
    '6':(1,0,1,1,1,1,1),
    '7':(1,1,1,0,0,0,0),
    '8':(1,1,1,1,1,1,1),
    '9':(1,1,1,1,0,1,1)}

try:
    while True:
        n = time.ctime()[11:13]+time.ctime()[14:16]
        s = str(n).rjust(4)
        for digit in range(4):
            for loop in range(0,7):
                GPIO.output(segments[loop], num[s[digit]][loop])
                if (int(time.ctime()[18:19])%2 == 0) and (digit == 1):
                    GPIO.output(25, 1)
                else:
                    GPIO.output(25, 0)
            GPIO.output(digits[digit], 0)
            time.sleep(0.001)
            GPIO.output(digits[digit], 1)
finally:
    GPIO.cleanup()
```