

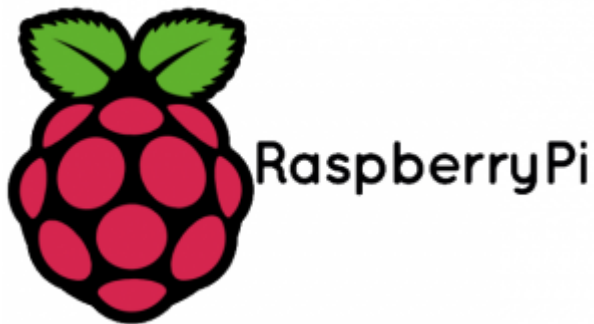
The DISTek Blog

≡ PRIMARY MENU

[Home](#) > **Part 2: Real-time on Raspberry pi**

Part 2: Real-time on Raspberry pi

POSTED ON OCTOBER 6, 2016 BY JOEL BENWAY



RaspberryPi

As promised, here's my Part Two blog posting regarding the raspberry pi.

The default kernel in [Raspbian Jesse](#), the latest release of the raspberry pi's official OS, could use a tune-up before being fit for duty running time-sensitive machine controls. On a vanilla Linux kernel program, latencies depend on everything running on the system making consistent and punctual tasks difficult to guarantee. The [RT PREEMPT patch](#) is a popular fix for this problem. The patch converts Linux into a fully preempt-able RTOS. If you want to control physical processes with your pi, as I intend to in DISTek's training curriculum update, this is the way to go. But how big of a difference does it make on a pi?

To answer this, I turned to the high resolution test program `cyclictst`. `Cyclictst` measures a system's latency response by running at regular intervals and measuring the time in between them. This simple algorithm accounts for everything that can delay a task and contribute to latency. Below is how you run this on the Raspberry pi:

```
pi@distekpi:~ $ git clone git://git.kernel.org/pub/scm/utis/rt-tests.git
```

```
pi@distekpi:~ $ cd rt-tests
```

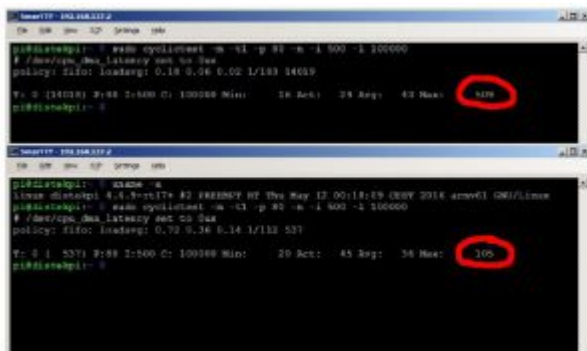
```
pi@distekpi:~ $ make all
```

```
pi@distekpi:~ $ cp ./cyclicttest /usr/bin/
```

```
pi@distekpi:~ $ sudo cyclicttest -m -t1 -p 80 -n -i 500 -l 100000
```

Note: on the newer multi-core raspberry pi's, the -t1 option should be -tn, where n is the number of cores.

Here are my before and after results on DISTek's Raspberry pi B+:



```
pi@distekpi:~ $ sudo cyclicttest -m -t1 -p 80 -n -i 500 -l 100000
$ /usr/bin/cyclicttest -m -t1 -p 80 -n -i 500 -l 100000
policy: fifo: loadavg: 0.18 0.36 0.02 1/183 34039
V: 0 [34039] F:88 2:540 C: 100188 Min: 18 Act: 38 Reg: 418 Max: 519
pi@distekpi:~ $
```

```
pi@distekpi:~ $ sudo cyclicttest -m -t1 -p 80 -n -i 500 -l 100000
$ /usr/bin/cyclicttest -m -t1 -p 80 -n -i 500 -l 100000
policy: fifo: loadavg: 0.79 0.36 0.14 1/183 537
V: 0 [ 537] F:88 2:540 C: 100188 Min: 39 Act: 45 Reg: 34 Max: 105
pi@distekpi:~ $
```

All of those times are in μ s. With this performance, 1ms tasks should be reliable and safe. For instructions on how to build and install a patched kernel to your pi, follow [this tutorial](https://www.distek.com/blog/part-2-real-time-on-raspberry-pi/) on the real time Linux wiki.

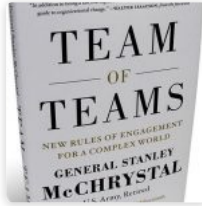
More from my site



Part 1: Cross-compiling for Raspberry pi



Integrated Projects Engineering Overview



Team of Teams



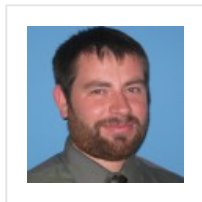
Out of the Rat's Nest: Why we Designed a Software-Defined I/O Simulator



Debugging: The Little Things



Despised by Engineers: Software Documentation



About Joel Benway

Joel has been a software engineer with DISTek since 2012. He has experience constructing and testing embedded software through Model Based Software Development (MBSD), C, and C++. He has a BSEE from the University of Wisconsin Milwaukee.



Share this post with your friends.....



Follow DISTek.....



POSTED IN EMBEDDED

TAGGED CYCLICTEST , LINUX KERNEL , RASPBERRY PI , RASPBIAN JESSE

DISTek takes on Farm Progress Show 2016

Fall 2016 ISOBUS Plugfest Review