PiCopter

Objective:

1) In this project, to get further close to an Unmanned Aerial Vehicle, it is vital to provide the top raspberry pi more space to run an advanced program to run achieve unmanned fly and move the controlling program from the top raspberry pi to the middle overseer.

2) Test case of the board as a raspberry pi shield.

Explanation of previous system

--xxxxxx

Design/Test plan (V model)

1. Peripheral driver level design/verification on bare-metal TI processor

1) I2C driver -- Sensor data fetching

2) General Purpose Timer -- (Use the right timer and priority)

2. Computational level verification

1) Sensor Fusion -- get rid of Linux dependencies

2) Printf support

3. System level design/verification

1) developing two run modes of program (Standalone by RF controller, and Pi-controlled) and changing SPI communication for support the switching mode feature as well as sensor data fetching.

2) To verify it

1. I2C development

1) understand how I2C works, reading standard NXP spec

2) understand how I2C driver in matt's code work (SMBus), clarify what I2C support I got from TI drivers (Generic I2C hardware module)

3) write a SMBus driver present at I2C protocol

2. General purpose timer

1) try to find an example (Tiva-C)

2) debug code by reading through TI timer module spec, register description

3) time generated by timer module is not right --- found a bug in TI API

3. Sensor fusion

1) start with removing Linux dependency

2) no printf support--- added in

3) start to test code using math lib--- found a bug

4) start to generate a reference from raspberry pi and compare the test result generated from the same piece of code and test input (lots of checking points had been set and using diff to compare them)

5) find a run-time bug due to unintelligence of TI compiler

4. Printf support

1) SPI solution

2) UART solution

3) JTAG solution --- found a wiki page of printf support on one of TI DSP.

5. Others: Major run-time bug/error

1) different behaviors generated from the same piece of source code by different compilers

2) processor is unable to run any user code and always fall into a fault ISR due to insufficient memory allocation in heap size and stack size

Goal achieved:

1) software finished but hardware has not been tested

2) provided more peripheral drivers as support for this board

3) built a test environment for self-checking (mainly for peripheral drivers)

4) created lots of specs and wiki-like docs/reference for further projects

The good:

1. I LIKE IT :) I want to have such a software experience, being exposed to new language, i.e. object-oriented language C++ (better understanding of low level language as well as OO) register-level operations on hardware module/processor, poor/beginner-unfriendly support/documentation. --- real challenges that an application engineer will have during working in a group

2. Create lots of documentations and wiki-like docs for whom this project might be passed to. --- what an application engineer would do during working in a group

3. GitHub --- using revision control tools to help manage the software project better and leave a log of the progress along this year. Created tags along the project. – what a software-related engineer would do during working.

The bad:

1. Sharing a JTAG programmer with another student in the first semester.

2. No documentation on software

3. No documentation on setting up TI eclipse project/Raspberry Pi eclipse project/ Quadcopter Implementation/Specification, and any wiki-like docs about this project, even just a weblink.

(even now, I don’t know how to set up a clean project from scratch for this project. I only knew a few settings that Matt has done about TI ARM compiler, linker and debugger, and having no awareness of these used to stop me from finding the bug)

The ugly:

1. No reference and little comments in his code, plus Matt was not around. The code he passed down to me was more like a released version rather than a draft version. (mainly because this is his 4th version of this piece of code)

2. No user online support on this TI processor (Tiva-C) and poor TI official support (The web page of this processor only listed some critical support on this processor, like very low-level driver, I2C, GPIO) (Only almost in the end of this project, I found lots of example codes in a software bundle by browsing the bundle and no any information on official/unofficial web directs me as a TI newbie to there)

3. Run-time error/bug when code was moved to another platform.