Fachhochschule Nordwestschweiz

Dokumentation

Konfiguration Cross-Compiling

Konfiguration um C++-Programme für den Phytec-SBC zu kompilieren

17. Oktober 2016

Saner Kevin

Institut für Automation

1 Ubuntu

Als Host-Computer wird idealerweise ein Computer mit Ubuntu 14.04 oder einem ähnlichen Betriebssystem eingesetzt. Wird ein dabei ein SBC von Phytec eingesetzt empfiehlt es sich alle Arbeitsschritte mit dem mitgelieferten Betriebssystem durchzuführen. Dieses verfügt schon über die entsprechende Toolchain. Wird dabei ein eigens aufgesetztes OS eingesetzt muss diese zuerst noch kompiliert werden.

2 Cross-Compiling

Da der Phytec Single Board Computer über eine ARM-Architektur verfügt muss das Programm speziell dafür kompiliert werden. Alle benötigten Tools für das Cross-Compiling sind dabei auf dem mitgelieferten USB-Stick. Alternativ kann der Kernel und die entsprechende Toolchain auch selbst kompiliert werden. Der Phytec SBC setzt dabei auf das Betriebssystem Yocto.

2.1 Eclipse

Da die Entwicklungsumgebung Eclipse bereits auf dem Phytec USB-Stick vorhanden ist, empfiehlt es diese auch zu verwenden. Eine Anleitung wie diese für das Entwickeln von C-Programmen konfiguriert werden muss, befindet sich ebenfalls auf dem USB-Stick (Application Guide). Um zu überprüfen, ob die Toolchain und der Compiler korrekt funktionieren sollte zuerst ein einfaches C-Programm erstellt werden. Eine komplette Anleitung um ein „Hello World“ in C zu entwickeln findet man dabei im „Application Guide“ (S. 10 - 23 [1]) von Phytec. Um für verschieden Architekturen zu kompilieren können in Eclipse verschieden Build-Konfigurationen hinterlegt werden, so kann das Debuggen lokal durchgeführt werden.

2.2 C++-Compiling

Das Mavlink-Interface wie auch das Gimbal-Interface wurden dabei in C++ umgesetzt, was es ein wenig komplizierter macht dieses zu kompilieren. Die folgende Anleitung soll Schritt für Schritt aufzeigen was unternommen werden muss, um ein auf dem Phytec lauffähiges Executable eines C++-Programms zu erhalten.

Projekt erstellen:
Für die Entwicklung eines C++-Programms muss in Eclipse ein neues C++-Projekt erstellt und benannt werden. Ein leeres Projekt ist dabei am besten geeignet.

Build-Pfade (*Mavlink):*
Damit für das Mavlink-Interface sämtliche Header-Files zur Verfügung stehen, muss das

entsprechende Verzeichnis im Projekt miteinbezogen werden. Mit „rechts-klicken“ auf das Projekt können unter „Properties -> C/C++Build -> Settings -> Includes“ Build-Pfade zum Projekt hinzugefügt werden. Für das Mavlink-Interface muss das Verzeichnis „/path/to/dir/mavlink/include/mavlink/v1.0“ hinzugefügt werden.

Für das Programm „Gimbal_control“ muss kein Build-Pfad angegeben werden.

C++- und Header-Files:
Nun kann das Entwickeln der Software beginnen. Stehen die entsprechenden C++- wie auch deren Header-Files bereits zur Verfügung können diese nun zum Projekt hinzugefügt werden.

Konfiguration der Toolchain:

Mavlink-Spezifisch: *Da das Mavlink-Interface-Programm ein sogenanntes „multithreaded“ Programm ist, muss die Bibliothek „pthread“ ins Programm eingebunden werden. Dazu muss unter „Properties -> C/C++ Build -> Settings -> GCC C++ Linker -> Libraries“ „pthread“ hinzugefügt werden.*

Kompiler
Die Konfiguration des „GCC C Compiler“ erfolgt gleich wie zur Entwicklung von C-Programmen. Als „Command“ muss lediglich ${CC} eingegeben werden. Auf die gleiche Art kann auch der „GCC C++ Compiler“ konfiguriert werden.

Linker
Als Linker kann jedoch nicht der GCC-Linker verwendet werden. Dazu muss ein G++-Linker verwendet werden. Dazu muss der komplette Pfad des Linker eingebunden werden. Somit muss zuerst nach dem G++-Compiler gesucht werden. Das gesuchte File heisst „arm-phytec-linux-gnueabi-g++“ und befindet sich normalerweise im Ordner „/opt/yogurt/iMX6-PD15.2.0/sysroot/x86_64-yogurtsdk-linux/usr/bin/arm-phytec-gnueabi“. Wie im Application Guide (siehe [1]) beschrieben muss auch hier in den Linker-Settings ${LDFLAGS} ergänzt werden, zudem muss vor ${LDFLAGS} ein Abstand gelassen werden.

Assembler
Als letztes muss auch noch der „GCC Assembler“ konfiguriert werden. Dazu muss auch lediglich unter „Command“ ${AS} eingegeben werden.

Nun sollte durch „Build Project“ erfolgreich ein Binary (Executable) hergestellt werden können. Die oben aufgeführten Schritte sind auch im Application Guide (siehe [1]) zum Phyboard Mira ersichtlich. Allerdings wird darin lediglich gezeigt wie C-Programme kompiliert werden. Der Unterschied der Konfiguration von C zu C++ ist dabei lediglich die Linker-Konfiguration.

Post-build (optional):
Um das Programm sogleich auf dem Phytec ausführen zu können, können die Post-build steps so konfiguriert werden, dass das erstellt Executable sogleich auf den Phytec geladen und ausgeführt wird. Dazu muss folgendes Command ergänzt werden: „scp ./NameOfExec root@192.168.3.11:/. ;ssh root@192.168.3.11 /NameOfExec“. Das Executable wird so im Root-Verzeichnis des SBCs ausgeführt.

Achtung: Die Netzwerkumgebung muss natürlich zuerst richtig konfiguriert werden, damit die Post-build steps korrekt ausgeführt werden.

Konfiguration Phytec
Wird zum ersten Mal versucht das erstellte Executable auf dem SBC auszuführen, wird wahrscheinlich eine Fehlermeldung angezeigt. Nämlich, dass das File nicht gefunden wird. Diese Fehlermeldung wird ausgegeben da der zum Ausführen des Programms benötigte Interpreter nicht gefunden wird. Der Name des benötigten Interpreters erfährt man durch die folgende Befehlszeileneingabe:

Command: $ readelf -l NameOfExec

Damit das Programm ausgeführt werden kann, muss das in der Zeile „Requesting program interpreter“ File im richtigen Verzeichnis vorhanden sein. Sucht man dieses File auf dem Phytec SBC wird man feststellen, dass dieses nicht vorhanden ist. Allerdings existiert ein File mit einem ähnlichen Namen, wie zum Beispiel „ld-linux-armhf.so.3“. Dies ist der auf dem Phytec vorhandene Interpreter oder zumindest zeigt dieses File auf den Interpreter. Um nun den Interpreter dem erstellten Programm zugänglich zu machen muss ein symbolischer Link (siehe [2]) erstellt werden, der auf das File „ld-linux-armhf.so.3“ zeigt. Der Link kann wie folgt erstellt werden:

Command: $ ln -s /lib/ld-linux-armhf.so.3 /lib/ld-linux.so.3

Nun sollte das erstellte Programm ausgeführt werden können. Dieser Schritt muss auf dem Phytec SBC lediglich einmal durchgeführt werden. Der momentan eingesetzte SBC wurde bereits konfiguriert, wird dieser jedoch einmal ersetzt oder tritt der oben beschriebene Fehler auf, müssen die entsprechenden Schritte durchgeführt werden.

Literaturverzeichnis

[1] Phytec: *Application Guide* [online] Available at: http://www.phytec.de/fileadmin/user_upload/images/content/1.Products/SBCs/phyBOARD-Mira_i.MX6/L-806e_1.pdf [Zugriff am 28.02.2017]

[2] Wikipedia: *Symbolische Verknüpfung* [online] Available at: https://de.wikipedia.org/wiki/Symbolische_Verkn%C3%BCpfung [Zugriff am 28.02.2017]