TOPPERS 活用アイデア・アプリケーション開発コンテスト

部門　　：　~~活用アイデア部門~~
　　アプリケーション開発部門

作品のタイトル　　：　Docker を利用した TOPPERS BASE PLATFORM 向けビルド環境およびシミュレータによる組込み OS セミナー向けカーネルの開発

作成者　　：　アライブビジョンソフトウェア株式会社　髙橋和浩

共同作業者　　：　-

対象者　　：　プログラムを覚えたての初級者
　　RTOS/TOPPERS に触れてみたいが時間が無い方
　　TOPPERS カーネルをビルドに失敗した方

使用する開発成果物　　：

１．TOPPERS BASE PLATFORM STM32F401RE nucleo-64 向けの
　①環境　Docker イメージ
　②ビルド用　DockerFile
　③ARM マシン用の環境　Docker イメージ
　④ARM マシン用ビルド用 DockerFile

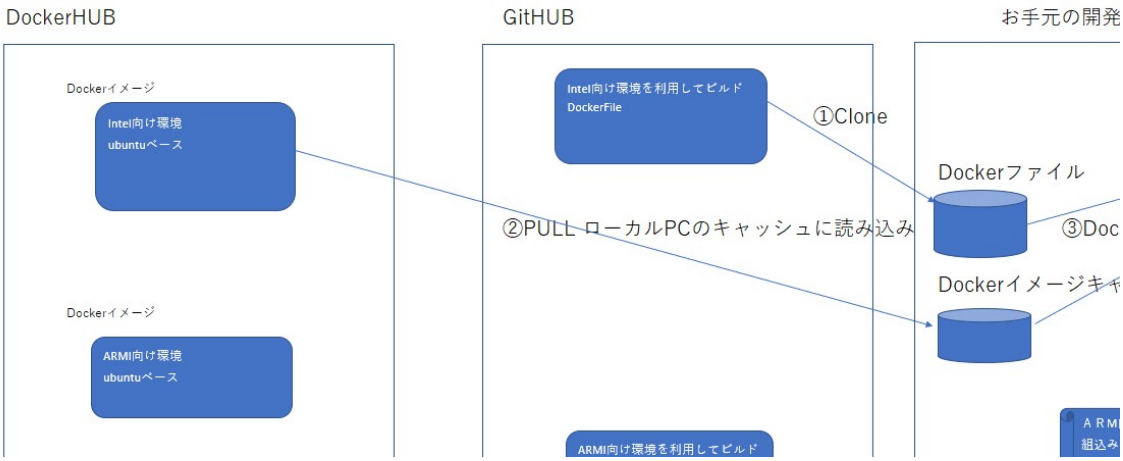
２.組込み OS セミナー用カーネル
　①カーネル本体（Visual Studio Code 用の設定ファイルが含まれている。）
　　環境　Docer イメージは　１.①③と同じ

３．添付資料「1 章クイックスタート.pdf」

目的・狙い

ARM マイコンが TOPPERS カーネルで利用されることが多く、開発環境と TOPPERS カーネルのビルドにおいて、初見者にとってハードルになるケースがあることを一部のユーザーからご意見をいただいています。
ARM のツールチェーンおよび、TOPPERS 特有のツール(コンフィギュレータ)
さらには、QEMU の環境や実機であれば TOPPERS BASE PLATFORM の関連ツールなどが
一発でビルドできれば解決できるのではと考えました。　さらに個々人があれこれ設定することや個別の linux 環境の差異による環境構築が上手くいかない問題ができるだけ最小にできればと考えました。
今回は Docker しかも　Docker のビルド機能のみで環境差異を最小にした状況での TOPPERS カーネルのビルドを行える環境を用意しました。

アイデア/アプリケーションの概要



上記がビルド環境のイメージ図です。
構成として以下の通りです。
① DockerHUB にインストール済みビルドツールイメージ
② TOPPERS を②の環境でビルドする DockerFile を GitHUB に
③ ホスト PC のアーキテキチャ別にご用意しています。
実行方法の概要
① GitHUB から DockerFile を Clone します。
② DockerHUB から pull しておきます。
③ Docker ビルドでカーネルビルドまで終了します。
Docker run して中身を確認します。
また組込み OS セミナー用カーネルを本環境に適用し、ベンチャー企業にてセミナーを実施しました。
弊社ホームページ内　以下の URL
http://alivevision.o.oo7.jp/rtos-seminor-report.html

アプリケーションの詳細

提供物の仕様（特徴）

対応 OS および確認マシン

Windows10 64bit 8GB メモリ　　空きストレージ容量 20GB 以上

確認マシン　ThinkPad X240 第４世代　Corei3

ChromeOS　AMD64 4GB メモリ　空きストレージ容量 20GB 以上

確認マシン　ASUS 425TA 4GB Core m3

ChromeOS　ARM64 4GB メモリ　空きストレージ容量 20GB 以上

確認マシン　Lenovo IdeaPad Duet Chromebook 4GB メモリ
128GB ストレージ

対応カーネル

TOPPERS/ASP　Intel/ARM 両方

TOPPERS/ASP3　ARM のみ

ASP3 について Intel 環境はありませんが、方式は同じなので、DockerFile からコピーして追加すれば可能と考えます。

また　Windows の Docker Desktop では、クロスプラットフォームの対応なので Windows でも時間がかかりますが実行可能です。

対応ターゲット

シミュレータ	ARM QEMU zynq および　Raspberry-pi2
実機	Raspberry-pi2
実機	TOPPERS BASE PLATFORM STM32F401RE nucleo-64

Docker イメージの仕様

AMD64 用 Docker イメージ

toppersjp/armgcc-ubuntu:7-2018-q2 をベース

QEMU はベースに内蔵

Intel32bit バイナリ実行環境を追加　コンフュギュレータ用

ただしコンフィギュレータは内蔵していない。

ARM64 用 Docker イメージ

ubuntu:20.04 をベース

QEMU は toppers/qemu_zynq.git のソースからビルドしている
コンフュギュレータは、DockerFile の github に ARM64 でビルドした
バイナリを同梱している。

各成果物のありか(URL)

1.①環境　Docker イメージ
https://hub.docker.com/repository/docker/alvstakahashi/stm32toolchain

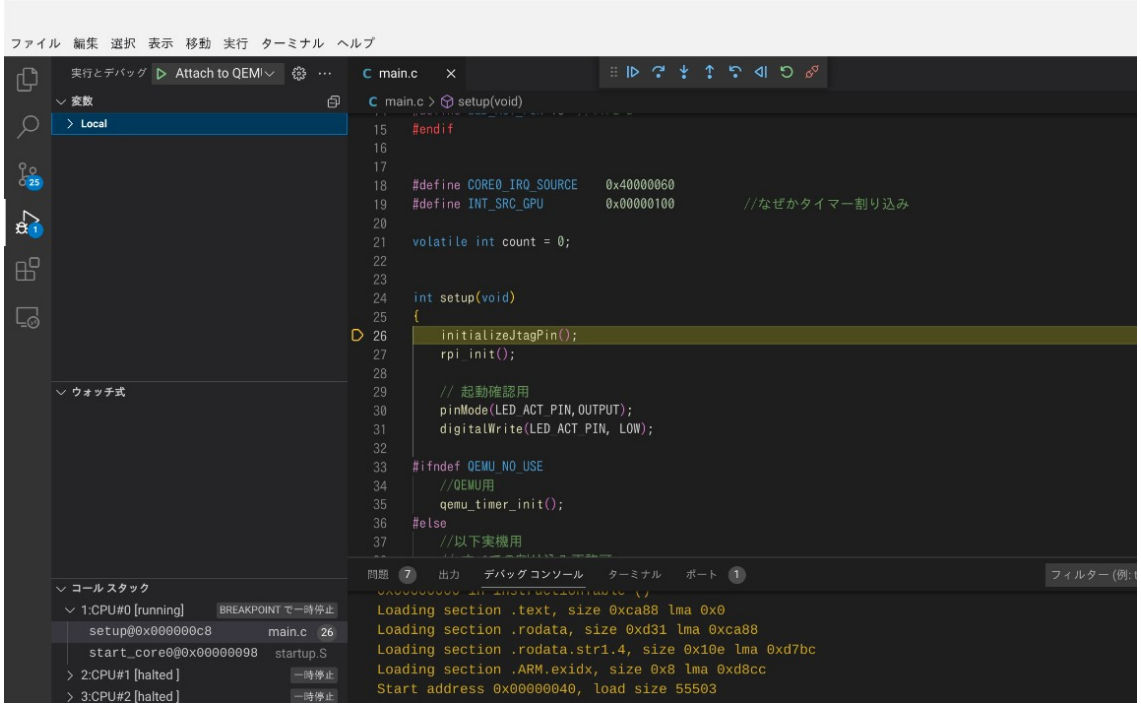
これを生成した DockerFile は右の通り
https://github.com/alvstakahashi/stm32toolchain

1.②ビルド用　DockerFile
https://github.com/alvstakahashi/Toppers_ASP_Build_STM32
1.③ARM マシン用の環境　Docker イメージ
https://hub.docker.com/repository/docker/alvstakahashi/stmtoolchainforduet

これを生成した DockerFile は右の通り
https://github.com/alvstakahashi/stmtoolchainforduet/blob/main/Dockerfile
1.④ARM マシン用　DockerFile
https://github.com/alvstakahashi/Toppers_ASP3_Build_STM32_Duet/blob/main/Dockerfile
2.①カーネル本体
https://github.com/alvstakahashi/RPI-SHRINK-SSP-FULL/tree/forRPI3-JTAG

実行方法

Windows 環境においては初級ユーザ向けに PC の仮想化 BIOS の設定から組込み OS セミナーのビルド実行までを図示したマニュアルをベースに順次進めていただければ実行可能です。添付資料「1 章クイックスタート.pdf」
手順通り実施すれば以下のコードレベルデバッガまで確認できます。



（このスクリーンキャプチャは　ARM64　ChromeOS 上で実行した例です）

ChromeOS での環境設定については、未作成です。ネットの情報等で確認し
①　Docker のインストール　②Visual Studio Code　のインストール　を行ってください。

Visual Studio Code　まで設定済みでえあれば、次の手順にて確認が可能です。

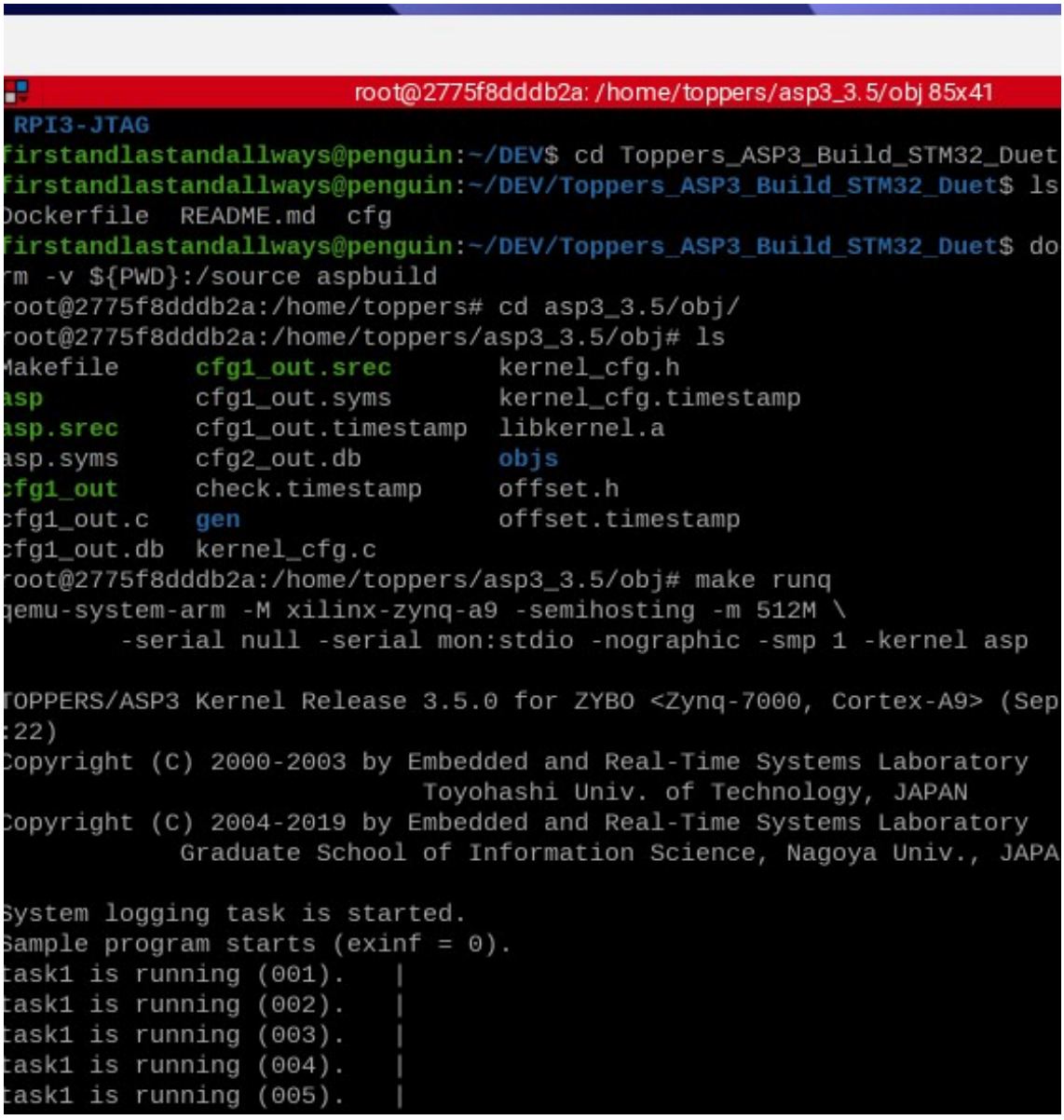
【実行手順】
1.TOPPERS BASE PLATFORM STM32F401RE nucleo-64 TOPPERS/ASP をビルドする場合

①ネットワークが重い場合があるので Docker イメージをローカルに置いておく
power シェルで実行します
> docker pull alvstakahashi/stm32toolchain:1
受信できているか確認します。
>docker images

②ubuntu ターミナルでシェルから実行する。Git から　Clone する。
適当なフォルダに移動する。以下　c:¥DEV¥ASP の例
$ cd /mnt/c/DEV/ASP
$ git clone https://github.com/alvstakahashi/Toppers_ASP_Build_STM32.git

③Docker ビルドを実行
$ cd Toppers_ASP_Build
$ Docker build -t aspbuild .
④できてることを確認する
Power シェルで確認
>docker run -it --rm -v ${PWD}:/source aspbuild
実行するとシェルが立ち上がっています。
　root@e25ce2665f1b:/home/toppers#
ビルドファイルの確認します
　　root@e25ce2665f1b:/home/toppers# ls asp/OBJ
　　.o ファイル　asp ファイルなどが表示されれば OK です。

⑤Windows の環境にコピーするには
df でマウント情報を確認すると、/source が先ほどのフォルダなので
root@e25ce2665f1b:/home/toppers# cp -r asp/* /source/.
でコピーできます。
2.それ以外の環境
それぞれのビルド用 DockerFile を公開している README.md を参照ください。
基本的に上記 1 と同じです。それぞれの①②③④⑤を実施してみてください。
以下　ARM64 環境でビルド後　Zynq マシン用のバイナリを QEMU で実行例になります。


root@2775f8dddb2a: /home/toppers/asp3_3.5/obj 85x41
RPI3-JTAG
firstandlastandallways@penguin:~/DEV$ cd Toppers_ASP3_Build_STM32_Duet
firstandlastandallways@penguin:~/DEV/Toppers_ASP3_Build_STM32_Duet$ ls
Dockerfile README.md cfg
firstandlastandallways@penguin:~/DEV/Toppers_ASP3_Build_STM32_Duet$ do
rm -v ${PWD}:/source aspbuild
root@2775f8dddb2a:/home/toppers# cd asp3_3.5/obj/
root@2775f8dddb2a:/home/toppers/asp3_3.5/obj# ls
Makefile cfg1_out.srec kernel_cfg.h
asp cfg1_out.syms kernel_cfg.timestamp
asp.srec cfg1_out.timestamp libkernel.a
asp.syms cfg2_out.db objs
cfg1_out check.timestamp offset.h
cfg1_out.c gen offset.timestamp
cfg1_out.db kernel_cfg.c
root@2775f8dddb2a:/home/toppers/asp3_3.5/obj# make runq
qemu-system-arm -M xilinx-zynq-a9 -semihosting -m 512M \
-serial null -serial mon:stdio -nographic -smp 1 -kernel asp
TOPPERS/ASP3 Kernel Release 3.5.0 for ZYBO <Zynq-7000, Cortex-A9> (Sep
:22)
Copyright (C) 2000-2003 by Embedded and Real-Time Systems Laboratory
Toyohashi Univ. of Technology, JAPAN
Copyright (C) 2004-2019 by Embedded and Real-Time Systems Laboratory
Graduate School of Information Science, Nagoya Univ., JAPA
System logging task is started.
Sample program starts (exinf = 0).
task1 is running (001). |
task1 is running (002). |
task1 is running (003). |
task1 is running (004). |
task1 is running (005). |

以上