Using the Yocto Autobuilder for Build and Release Management

Jate Sujjavanich
Syntech Systems, Inc
<jatedev -at- gmail.com>

February 22, 2016

My Experience

- PCB/Schematic Design
- Das U-boot
- uClinux-dist
- Linux Kernel Drivers
- Open Source Tools
- Yocto Project Tools: poky, bitbake, etc.

Talk Based on Dora Version

- Still using autobuilder from June 2014
- Technical debt due to priorities
- Bug fixes have been backported

Ian Gampon
https://creativecommons.org/licenses/by/2.0/

Build and Release Management

Why Autobuilder?

Complex code requires continuous integration

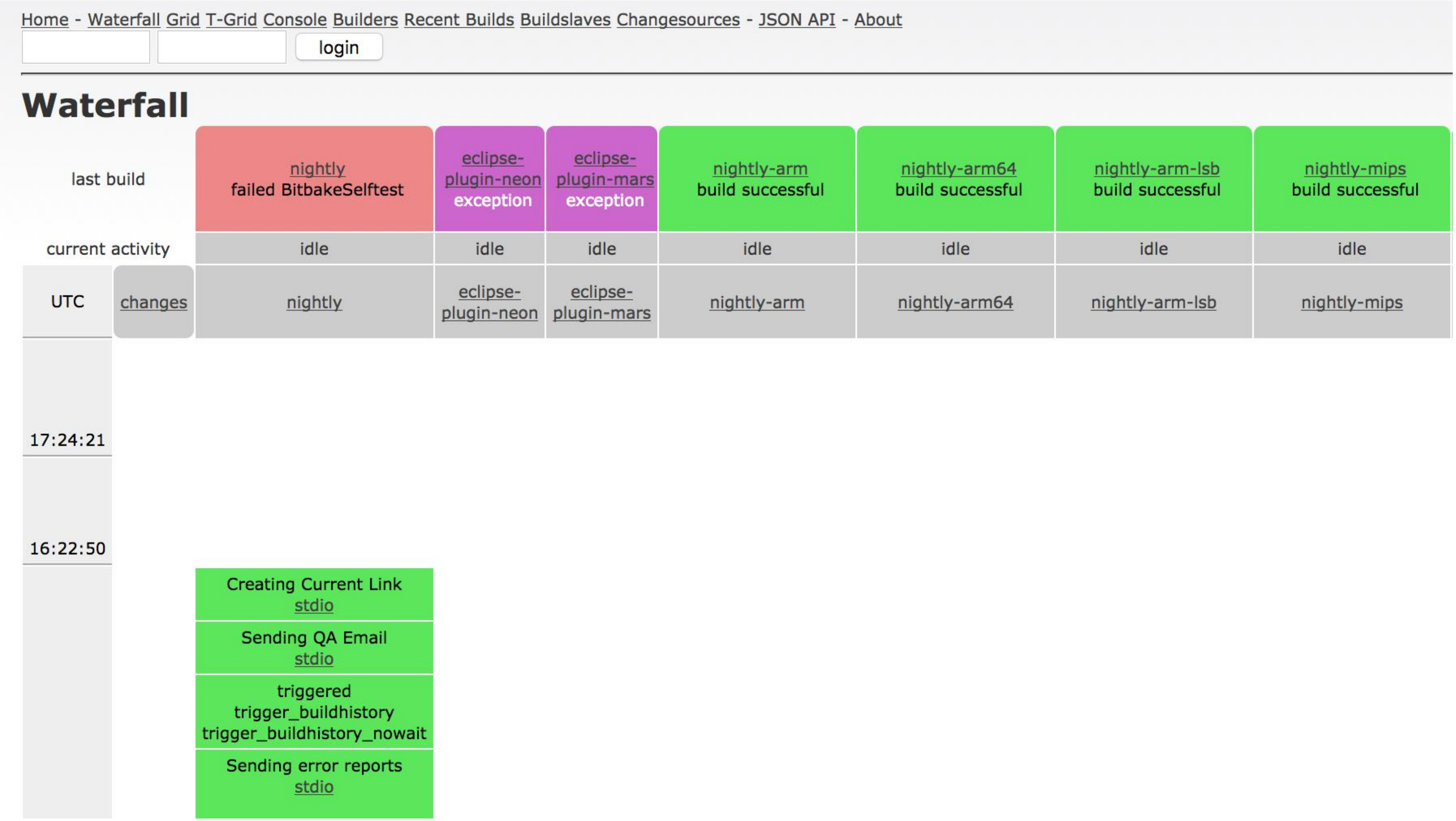
- Many opportunities for mistakes
  - local.conf
  - Multiple layers
  - .bbappend’s

You need official, clean builds of your source

Yocto Project’s Autobuilder

- Buildbot adapted for use with the poky/oe-core
- Layer Retrieval
- Buildsteps for
  - Configuration files
  - Bitbake targets
  - Publishing artifacts

autobuilder.yoctoproject.org



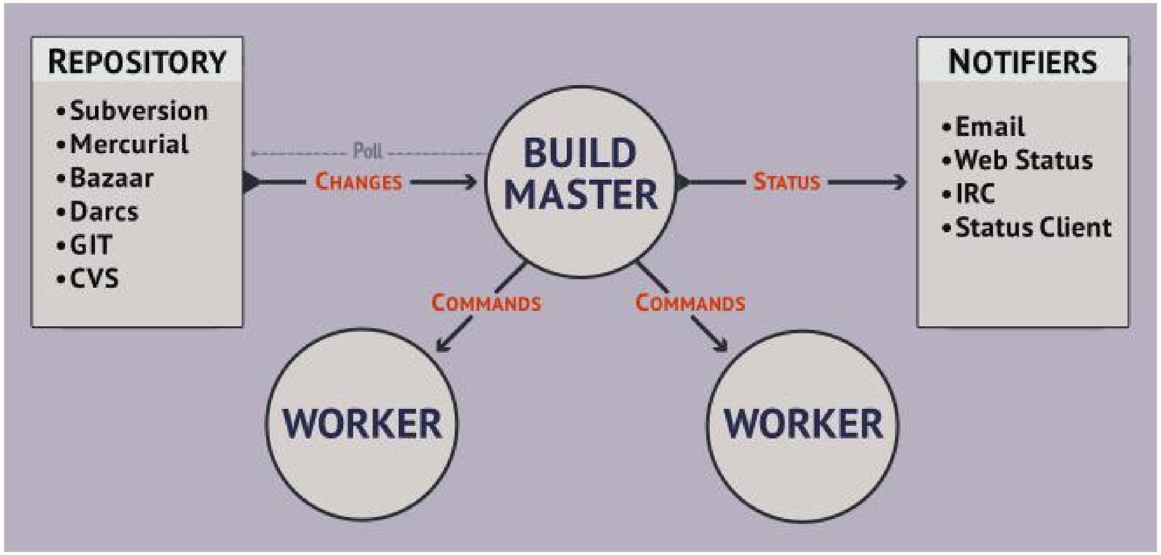
Home - Waterfall Grid T-Grid Console Builders Recent Builds Buildslaves Changesources - JSON API - About

login

Waterfall

last build		nightly failed BitbakeSelftest	eclipse-plugin-neon exception	eclipse-plugin-mars exception	nightly-arm build successful	nightly-arm64 build successful	nightly-arm-lsb build successful	nightly-mips build successful
current activity		idle	idle	idle	idle	idle	idle	idle
UTC	changes	nightly	eclipse-plugin-neon	eclipse-plugin-mars	nightly-arm	nightly-arm64	nightly-arm-lsb	nightly-mips
17:24:21								
16:22:50								
		Creating Current Link stdio						
		Sending QA Email stdio						
		triggered trigger_buildhistory trigger_buildhistory_nowait						
		Sending error reports stdio						

Controller and Workers



www.buildbot.net

autobuilder.yoctoproject.org

- One Controller commands Nine Workers

Our smaller scale project has 1:1

What is a job

title

```
[nightly-fsl-arm]
builders: 'example-worker'
repos: [{'poky':
            {'repourl':'git://git.yoctoproject.org/poky',
             'layerversion':{'core':'meta', 'yoctobsp':'meta-yocto-bsp', 'yocto':'meta-yocto', 'poky':'meta-poky'},
             'branch':'master'}},
 <SNIP>
steps: [{'SetDest':{}},
        {'CheckOutLayers': {}},
        {'RunPreamble': {}},
        {'GetDistroVersion' : {'distro': 'poky'}},
        {'CreateAutoConf': {'machine': 'imx53qsb', 'SDKMACHINE' : 'x86_64',
 <SNIP>
        {'BuildImages': {'images': 'core-image-sato core-image-sato-dev core-image-sato-sdk core-image-minimal core-image-minimal-dev'}},
scheduler: [scheduler: [{'nightly-scheduler' :
                {'type':'Nightly', 'hour':'2', 'minute':'0',}
```

repos

buildsteps

scheduler

autobuilder.yoctoproject.org

buildset-config.qa

23 example jobs on the official

nightly-arm.conf	nightly-world.conf	nightly-qa-pam.conf
nightly-deb.conf	nightly-x32.conf	nightly-qa-skeleton.conf
nightly-ipk.conf	nightly-x86.conf	nightly-qa-systemd.conf
nightly-rpm.conf	poky-tiny.conf	nightly-qa-targetbuilds.conf
nightly-multilib.conf	nightly-x86-64.conf	buildtools.conf
nightly-oe-build-perf-test.conf	nightly-qa-distro.conf	eclipse-plugin-neon.conf
nightly-oe-selftest.conf	nightly-qa-extras.conf	
nightly-oecore.conf	nightly-qa-logrotate.conf	

images

qa

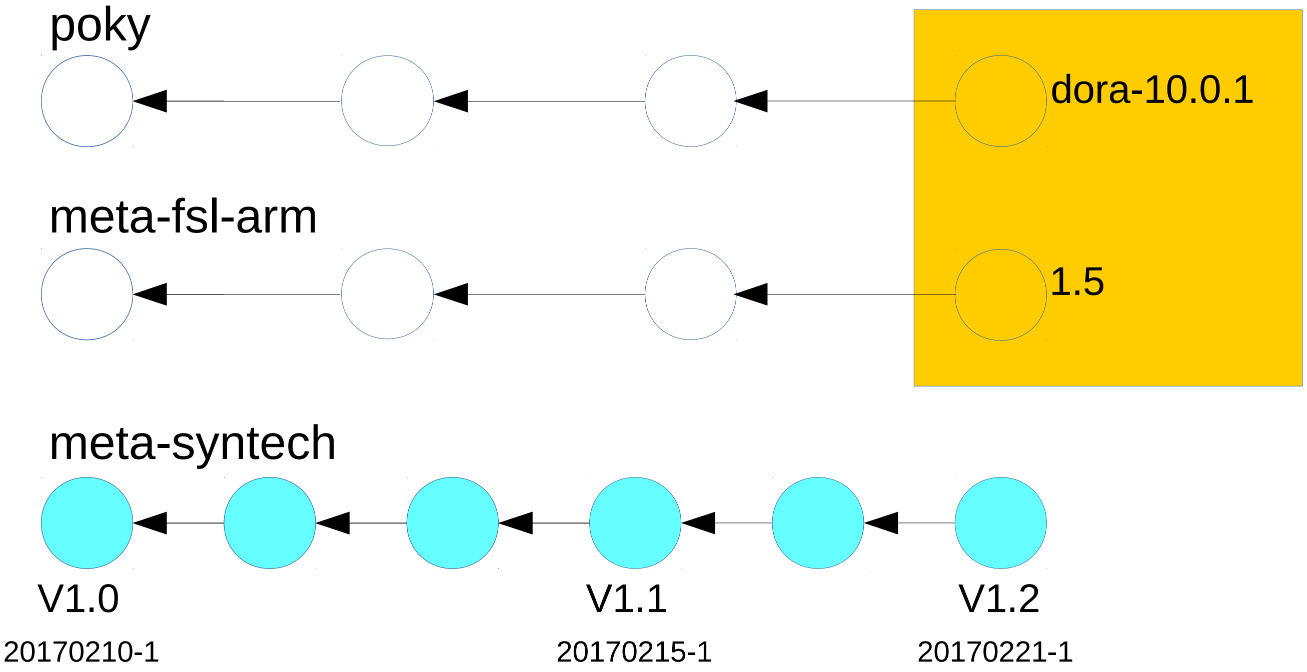
Build tools

Our Jobs

Create your own jobs based on source tree examples

- Nightly
  - Populate recipe downloads
- Qemux86
  - Sanity check
- Product image - master
- Product image - stable
- Project RPM Build

Repos: Combining Upstream and Local Layers



Job Example - “repos:”

- Upstream Layers (low delta)
  - Maintainers tie revisions to poky releases

```
{'meta-fsl-arm':
  {'repourl':
    'git://github.com/Freescale/meta-fsl-arm.git',
    'hash': 'af392c22bf6b563525ede4a81b6755ff1dd2c1c6' }},
```

- In-house Layers (high delta)

```
{'meta-syntech':
  {'repourl':'git://tux1/git/meta-syntech.git',
   'branch':'master'}}
```

Layer Checkout Scripts for Developers

- Developer Script
  - Matches repo section of autobuilder job
  - Clones fixed upstream and local HEAD
- Release Script
  - Saves SRCREV of local and upstream repos
  - Tracked in source control for reproducibility

Custom Buildstep:
ReleaseSyntech.py

Automates release activities

- Convert SRCREV from PublishLayerTarballs to Release Script
- Perform Release actions on local layer copies in builds
  - Create git tags (like v1.0) for reference
  - Commit release script
  - Image version bump (within recipe)
- Release Manager pushes repo’s if autobuilder artifacts are accepted

Using `TEMPLATECONF` for Build Configuration

- First step “. oe-init-build-env” pulls default bblayers.conf and local.conf
- Control local.conf and bblayers.conf in source control
- From dir poky:

```
TEMPLATECONF=../meta-syntech-bsp/conf . oe-init-build-env
```

- From layer specified in TEMPLATECONF:
  - local.conf.sample → build/conf/local.conf
  - bblayers.conf.sample → build/conf/bblayers.conf

TEMPLATECONF: Buildstep: RunPreamble

- RunPreamble
  - Calls . oe-init-build-env
- RunPreambleSyntech
  - Customizes by adding TEMPLATECONF
  - There is a new altcmd in RunPreamble (post dora)
- Note: auto.conf allows autobuilder specific config
  - Comes before settings in local.conf

PublishArtifacts.py

Bulk of the code is for the official autobuilder

```
for artifact in self.artifacts:
...
    if artifact == "adt-installer":
...
    elif "eclipse-plugin" in artifact:
...
else:
        command += self.generateMD5cmd(artifact, 
deploy_image_dir)
        if "beagle" in artifact:
```

PublishArtifactsSyntech.py

- Much of the Yocto code is N/A to our needs
- Deploys everything in deploy/images by default
  - `core-image-sato-mx6sabresd.ext3 → core-image-sato-mx6sabresd-20170217152000.ext3`
- For release, we only use a subset of the build artifacts
  - U-boot binary
  - Device tree
  - Linux Kernel
  - Specific File System Image

Buildstep Coding Tips

Mapping Config to Buildsteps

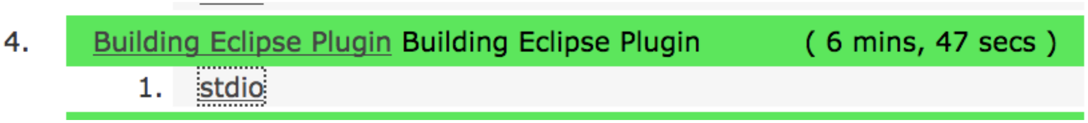
- Code Path from Environment to Python
  - config/autobuilder.conf
  - autobuilder/buildset.py (converts ENV to python)
  - buildsteps/SomeAction.py
- yocto-autobuilder-setup matches many searches, but it’s all placeholders

Buildstep Coding Tips
The Shell Code Within

- Typical buildstep shell code

```
command = “cd “ + self.workdir + “;”
command += “cp * “ + self.output + “;”
```

- Output is viewable on autobuilder buildstep logs



4. Building Eclipse Plugin Building Eclipse Plugin (6 mins, 47 secs)
   1. stdio

- Example output (off the browser screen):

```
cd /home/jate/yocto-autobuilder/yocto-worker/example ; cp *
```

Buildstep Coding Tips

The Shell Code Within

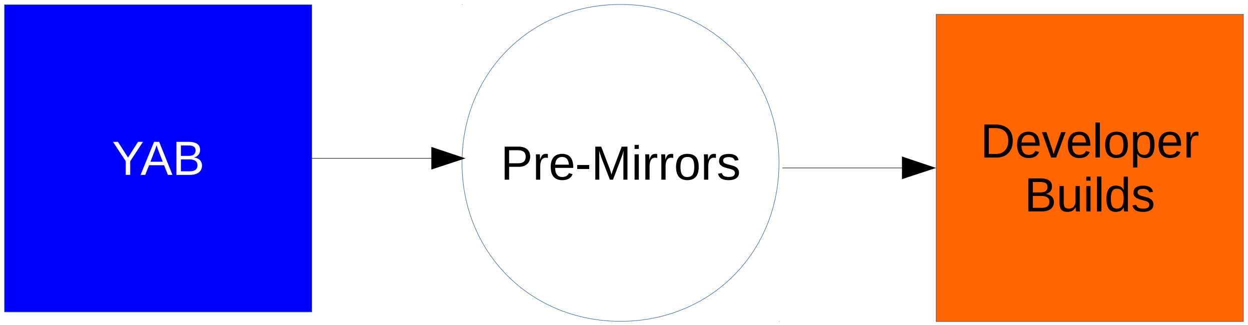
- Add carriage returns for readability

```
command = “cd “ + self.workdir + “\n”

command += “cp * “ + self.output + “\n”
```

- Improved output:

```
cd /home/jate/yocto-autobuilder/yocto-worker/example

cp * /tmp/yab/output
```

Speed Builds by Populating Premirrors with Yocto Autobuilder



- Pre-Mirror sstate_cache/downloads treated as read-only by developer builds
- Developer builds symlink to premirror first
- autobuilder.conf: Tell YAB to use premirror SSTATE_CACHE and DL_DIR
- Autobuilder jobs will populate the pre-mirrors

Autobuilder Maintenance
Disk Space

- Cronjob to clean

```
* 6 * * * clean.sh

find /tmp/publish/machine -maxdepth 1 -type d
-ctime +5d -exec rm -rf {} \;
```

- Cleaning of sstate-cache

```
sstate-cache-management.sh
```

- Protip: Windows does not support symbolic links, and makes full copies

Configuration Storage

Store your configuration in the autobuilder git tree

- buildset-config.syntech
- controller.cfg
  - Web GUI accounts
- autobuilder.conf
  - Environment variables

Tips & Tricks
Trigger Build Off Project GIT

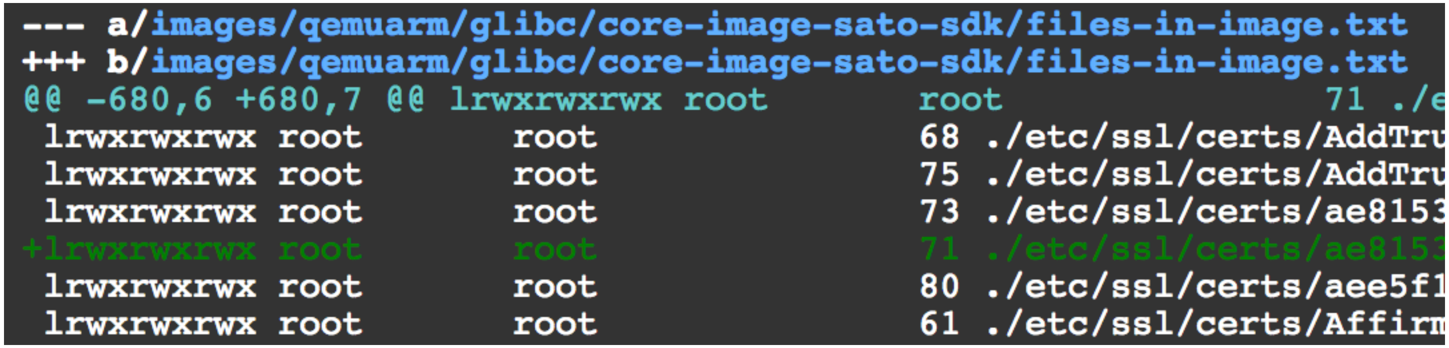
Nightly Image Builds Were Too Infrequent

- Added git-poller Scheduler (backported 4 commits)
- auto.conf hack: Change recipe to pull from the master of its SRC_URI

```
{‘CreateAutoConf’:{…

 ‘atextappend’: ...

‘SRCREV_pn-recipe = “${AUTOREV}”\n’

‘PV_append_pn-recipe = “+git${SRCPV}”\n’

(Bitbake order of variable flags)
```

Tips & Tricks
buildhistory

- Buildhistory bbclass outputs build data to a git repository
  - File name, sizes, modes
  - Package Data
- Add a tag for comparison between versions



```
--- a/images/qemuarm/glibc/core-image-sato-sdk/files-in-image.txt
+++ b/images/qemuarm/glibc/core-image-sato-sdk/files-in-image.txt
@@ -680,6 +680,7 @@ lrwxrwxrwx root       root               71 ./e
 lrwxrwxrwx root       root               68 ./etc/ssl/certs/AddTru
 lrwxrwxrwx root       root               75 ./etc/ssl/certs/AddTru
 lrwxrwxrwx root       root               73 ./etc/ssl/certs/ae8153
+lrwxrwxrwx root       root               71 ./etc/ssl/certs/ae8153
 lrwxrwxrwx root       root               80 ./etc/ssl/certs/aee5f1
 lrwxrwxrwx root       root               61 ./etc/ssl/certs/Affirm
```

Future Tasks

Deal With External Layer Outages

- External Layer Repositories
  - github.com/Freescale
  - git.freescale.com
  - git.yoctoproject.org
- Network Problems
  - Attack on DNS github.com
  - Local IT Issues

Future Tasks
Mirror for External Layers

- config/autobuilder.conf
  - OGIT_MIRROR_DIR
- ResolveLayers.py, Buildset.py
  - Did not support mirrors
- CheckOutLayers
  - `{'CheckOutLayers': {'clobberOnFailure': True}}`
    - `Updates, but hacked _fetch function to allow git fetch failure`
- Work In Progress
  - Method = fresh, mode = full?
  - Need to populate the mirror directory

Future Tasks

Combine “repos:” and checkout scripts

- Avoid errors converting between the two
- Simple conversion between “repo:” section and checkout script
- Utilize bitbake or buildbot yoctogit libraries

Future Tasks
PR Service

- Autobuilder ideal as the source for package feeds
- Run the service bitbake-prserv
- Specify PRSERV_HOST in auto.conf
- Back up database using
  - `bitbake-prserver-tool export`

Future Tasks
Automated Runtime Testing

- Mega Manual Section: Performing Automated Runtime Testing
- Runs QA tests like Yocto Project’s Autobuilder Does
- Dora: Python code executes commands on a QEMU instance
- Newer releases allow you to run on target hardware

Thank You

Questions?