Innexis 2025.1 Android Target Software Integration Instructions

This document only covers integrating SARC's GPU kernel and user mode drivers into the Innexis 2025.1 Android16 reference target software. It assumes that the SGPU RTL platform has already been integrated with the Innexis 2025.1 RTL reference platform.

Create updated super-qemu.img.qcow2

1. **Create a Clone of Android Using Innexis Installation**

   We will need clones of both Android15 and Android16

   ```
   # Replace ${INNEXIS_INSTALLATION} with the actual path where Innexis DevPro is installed
   innexis-clone ${INNEXIS_INSTALLATION}/platforms/hybrid/android15 <destination_path>
   innexis-clone ${INNEXIS_INSTALLATION}/platforms/hybrid/android16 <destination_path>
   ```

2. **Update config/site.sh and Source It**

   Open `config/site.sh` in a text editor and update environment:

   ```
   vi config/site.sh
   ```

   For example:

   - `INNEXIS_DEVPRO_HOME` will need to point to the current Innexis DevPro installation
   - `VLAB_HOME` will need to point to at least a 2025.1 version of VLAB
   - `SALT_LICENSE_SERVER` will need to be set

   Then source the script:

   ```
   source config/site.sh
   ```

   Do this for both the android15 and android16 clones.

   Note: The same `site.sh` can be used for both these clones.

3. **Navigate to Clone Directory and Run Extract Script**

   ```
   cd {./android-clone}/sw
   ./extract.sh
   ```

   Do this for both the android15 and android16 clones.

4. **Copy file_contexts.txt from android15 clone**

   The android16 clone does not have this, so we will copy it over from android15.

   ```
   cp ${ANDROID15_CLONE}/sw/output/host/custom_vendor/file_contexts.txt ${ANDROID16_CLONE}/sw/output/host/custom_vendor/
   ```

   Note: Once this step is complete, the android15 clone is no longer needed and can be removed.

5. **Copy and Edit prepare_super-qemu_env_innexis_2025-1_a16.bash**

   First, copy the script:

   ```
   cp /sarc-c/gpusw/users/Siemens_Run_Area/jcristob/scripts/prepare_super-qemu_env_innexis_2025-1_a16.bash .
   ```

   Open the script for editing:

   ```
   vi prepare_super-qemu_env_innexis_2025-1_a16.bash
   ```

   Update the variables as follows:

   - **CLONE_DIR**: Replace with the directory for your Android16 clone
   - **VENDOR_DIR**: Replace with path to where Samsung files are located
     - **Note:** VENDOR_DIR can likely be left unchanged

6. **Run the prepare_super-qemu_env_innexis_2025-1_a16.bash Script**

   This script will:

   - Patch file_contexts.txt and uevent.rc
   - Copy over Samsung files to their respective locations in custom_vendor/
   - Remove unnecessary files from Siemens in custom_vendor/

   ```
   ./prepare_super-qemu_env_innexis_2025-1_a16.bash
   ```

7. **Run the enable_hdlcd.sh script**

```
cd ${CLONE_DIR}/sw
./enable_hdlcd.sh
```

8. **Run the recombine script**

   Change directory and execute the script:

```
cd ${CLONE_DIR}/sw
./recombine.sh
```

The updated `super-qemu.img.qcow2` file will be located at:

```
${CLONE_DIR}/sw/output/images/super-qemu.img.qcow2
```

Update Kernel

1. **Make a copy of the kernel source**

```
cp -r /sarc-c/gpusw/users/Siemens_Run_Area/sgpu-kmd .
```

   Note: This is more of a workaround; I ran into permission issues when using git.

2. **Apply Kernel Patches**

   Apply patches to make sure your kernel is up-to-date and consistent.

   The patches can be found at `/sarc-c/gpusw/users/Siemens_Run_Area/jcristob/patches`

```
 # Replace ${sgpu-kmd_copy} with the path to your sgpu-kmd copy
# Replace ${PATCH} with the path to the patch
patch -g0 -p1 --no-backup-if-mismatch -d ${sgpu-kmd_copy} -t -N  < ${PATCH}
```

   - Do this for every patch in the directory
   - If a patch hunk fails, apply manually

3. **Update Symbolic Links**

   Modify symbolic links in `siemens-prebuilt` to point to the prebuilts in your clone directory.

```
cd ${sgpu-kmd_copy}/sgpu/sbuild/sbuild/sgpu/dependencies/siemens-prebuilt
ln -sf ${CLONE_DIR}/sw/prebuilt/* .
```

4. **Build the Kernel**

   Run the following commands to build the kernel:

```
cd ${sgpu-kmd_copy}
make olddefconfig
make ARCH=arm64 mrproper
./sgpu/build.py -d sgpu_hycon_android
```

   Final Images and Directories

   The final kernel image and ramdisk will be created in:

```
${sgpu-kmd_copy}/build/bare/arm64/release/sgpu_hycon_android/install/
```

Update dtb

1. **Copy dtb used for hycon runs and decompile it**

```
cp ${DTB_LOCATION}/sgpu-hycon-m4.dtb .
dtc -I dtb -O dts sgpu-hycon-m4.dtb > sgpu-innexis-m4.dts
```

   Note: A working hycon dtb can be found at: /sarc-c/gpusw/users/Siemens_Run_Area/Kernel_Feb7th_copy/sgpu-hycon-m4.dtb

2. **Edit the dts file**

```
vim sgpu-innexis-m4.dts
```

   The .dtb behavior has changed from Hycon to Innexis. Previously, Hycon would prioritize the .dtb generated by QEMU over the user's .dtb; however, Innexis

uses the memory map from the user's .dtb instead of the QEMU generated one. Thus, the memory entries will need to be updated to reflect the desired memory map.

1. Add the following two lines to the dts file, after the first `/memreserve/` line

```
/memreserve/  0x0000000084000000 0x0000000000048dc8;
/memreserve/  0x0000000083270000 0x00000000006ee0df;
```

2. Replace the `memory@80000000` block with the following:

```
memory@84048dc8 {
    reg = <0x0 0x0 0x0 0x1000000 0x0 0xe000000 0x0 0x1000000 0x0 0x80000000 0x0 \
    0x80000000 0x4 0x0 0x0 0x80000000 0x4 0x80000000 0x0 0x80000000>;
    device_type = "memory";
};
```

3. Replace the `interrupt-controller@2c001000` block with the following:

```
interrupt-controller@2c000000 {
    compatible = "arm,gic-v3";
    #interrupt-cells = <0x3>;
    #address-cells = <0x0>;
    ranges;
    interrupt-controller;
    reg = <0x0 0x2c000000 0x0 0x10000 0x0 0x2c040000 0x0 0x40000>;
    interrups = <0x1 0x9 0x4>;
    phandle = <0x1>;
};
```

Note: This last change is necessary because the hw platform has changed from GIC400 to GIC600.

3. **Recompile into dtb**

```
dtc -I dts -O dtb sgpu-innexis-m4.dts > sgpu-innexis-m4.dtb
```

Copy Build and Run

1. **Copy entire build directory**

```
cp -a ${INNEXIS_BUILD} .
```

Note: Copy the entire platform directory, not just 'veloce'

2. **Source site.sh**

```
source config/site.sh
```

3. **Edit platform-common.conf to point to your pieces**

```
vim config/generated/platform-common.conf
```

There are many places that will need updating in this file.

```
-drive "index=0,id=super-qemu.img.qcow2,...file=..."
```

- Replace "file..." with the super-qemu.img.qcow2 created in a previous step

```
-drive "index=1,id=userdata.img.qcow2,...file=..."
```

- Replace "file=..." with your userdata file:
  - ${ANDROID16_CLONE}/sw/output/images/userdata.img.qcow2
  - From the same clone where you created your super-qemu.img.qcow2

```
-dtb "..."
```

- Replace "..." with the path of the dtb file created in a previous step

```
-initrd "..."
```

- Replace "..." with the path of the ramdisk.img created in a previous step

```
-android-prop-file "..."
```

- Replace "..." with the path of the android-prop.config file
  - ${ANDROID16_CLONE}/sw/prebuild/android-prop.config
  - Use the same clone from which the super-qemu.img.qcow2 was created

```
elf_file : [ "..."]
```

- Replace "..." with the path of the Image file created in a previous step

4. **Run**

```
cd veloce
./run
```

Document version 2.0

Change Log:

[2.0] - 2025-06-13

Changed:

- Removed manual integration steps in "Update Kernel" section
- Created patches for said integration steps, and documented their usage
- Modified 'Build the Kernel' step to reflect new changes