TWL-SDK

About Components

Version 1.0.4

TWL-06-0050-001-B
Released: September 9, 2009

Table of Contents

Revision History

Version	Revision Date	Description
1.0.4	2009/06/15	Added text about the `ichneumon.TWL` component.
1.0.3	2008/12/05	Added support for the TWL-SDK.
1.0.2	2008/09/01	Made it clear in the overview that components cannot be dynamically switched while an application is executing.
1.0.1	2005/06/09	Revised text to reflect that the SND library is now included in the SDK. Corrected the location of programs on the ARM7 side to address `27e0000`.
1.0.0	2005/03/11	Initial version.

1 Overview of Components

1.1 Components in the TWL-SDK

In the TWL-SDK, the term “component” refers to a program that runs on the ARM7 processor (or “ARM7”). Because applications run on the ARM9 processor (or “ARM9”), components are mainly in charge of device control.

The ARM9 controls component operations through communications via a FIFO between the ARM9 and ARM7. The TWL-SDK provides a group of libraries that bring together the functions used to control the component for each device. As such, applications do not need to communicate directly with the ARM7 via the FIFO.

Although multiple components are provided with the TWL-SDK, only one must be selected for an application when the application is created. The single selected component runs on the ARM7, and a running application cannot switch between components.

1.2 Controlled Devices

This section describes the devices controlled by components and the ARM9 libraries provided to operate these devices.

1.2.1 Digital Buttons

With TWL hardware, the ARM9 cannot directly read the status of the X Button, Y Button, and open/close detection button. The component therefore monitors the status of these buttons. The PAD library provides functionality for reading the status of these buttons.

1.2.2 Sound Circuit

With TWL hardware, the ARM9 cannot directly access the sound output circuit. The component therefore controls the sound circuit. The SND library is provided as the ARM9-side library.

1.2.3 SPI Devices

With TWL hardware, the ARM9 cannot directly access SPI devices, and it cannot access more than a single SPI device at a time. The component therefore manages access to multiple SPI devices.

There are four SPI devices: the microphone, the touch panel, the power management IC, and the built-in flash memory.

For the microphone, the component manages and maintains an accurate sampling rate for continuous sampling. The MIC library provides microphone sampling functionality.

This component also manages and maintains an accurate sampling rate for continuous sampling of the touch panel. The TP library provides touch panel sampling functionality.

Various features that can be implemented using the power management IC are collected in the PM library.

As for the built-in flash memory, although components include flash memory read/write functionality, no API for controlling this feature from the ARM9 is provided. Support for reading data from the flash memory using the OS library is provided only for some save data such as owner information.

1.2.4 Real-Time Clock

With TWL hardware, the ARM9 cannot directly access the real-time clock. The component therefore controls the real-time clock. The RTC library provides the ability to control the real-time clock.

1.2.5 Wireless-Communication Module

With TWL hardware, the ARM9 cannot directly access the wireless-communication module. The component is responsible for various basic control functions related to wireless communications, such as control of the wireless-communication module, timing control required by the communication protocol, and automatic V-Blank synchronization with communication partners. With the ARM9, for NITRO wireless, the WM library and other libraries provide functionality for controlling wireless communications. For TWL wireless, the TWL-SDK Wi-Fi library provides functionality for controlling infrastructure communications.

1.2.6 Camera Module

With TWL hardware, the ARM9 cannot directly access the camera module, so the component controls the camera module. However, some features can only be accessed from the ARM9. The CAMERA library is provided as the ARM9-side library.

1.2.7 SD Card Interface

With TWL hardware, the ARM9 cannot directly access the SD card interface.

1.2.8 AES Module

With TWL hardware, the ARM9 cannot directly access the AES module, so the component controls the AES module. The AES library is provided as the ARM9-side library.

2 Component Types

Six types of components are provided with the TWL-SDK: `mongoose`, `ichneumon`, `mongoose.TWL`, `ichneumon.TWL`, `raccoon.TWL`, and `ferret.TWL`. A description of each of these components is given below.

2.1 mongoose

This is the default NITRO ROM component for controlling the five devices given in sections 1.2.1 through 1.2.5.

With this component, most of the executable code for features related to wireless communications is located in main memory. Main memory is therefore frequently accessed when executing communication-related processes. When the ARM7 is accessing main memory, main memory access from the ARM9 and access by DMA must wait. If the default main memory access priority setting is used, the time available for the ARM9 to access main memory decreases as the frequency of communications and the amount of communication data increases. In the worst case, there are times when the ARM9 can hardly access main memory at all for several picture frames when the ARM7 is performing a series of special operations, such as during the connection process.

2.2 ichneumon

Although this NITRO ROM component also controls the five devices given in sections 1.2.1 through 1.2.5, the portion of this component that controls wireless communications is special-purpose.

Unlike the `mongoose` component, with `ichneumon` most of the executable code for features related to wireless communications is located in VRAM. Compared to the `mongoose` component, this feature significantly decreases the frequency at which the ARM7 accesses main memory when executing communication-related processes.

When using the `ichneumon` component, at least one of VRAM-C or VRAM-D must be allocated to the ARM7 to allow the use of wireless-communication features. If neither of the VRAMs is allocated to the ARM7, an image of the program that should have been allocated to VRAM is backed up in the system-reserved area of main memory. The ARM9 can therefore use these VRAMs for other purposes if the wireless-communication features will not be used.

The Wireless Driver Operation Control (WVR) library is provided for enabling and disabling wireless-communication features by changing the VRAM allocation status.

2.3 mongoose.TWL

This component (the default for hybrid ROMs) can control all of the devices given in section 1.2 Controlled Devices. This component is an extension of `mongoose` for use with hybrid ROMs. However, when this component is running on NITRO hardware, it is limited to controlling the five devices given in sections 1.2.1 through 1.2.5.

2.4 ichneumon.TWL

Although this hybrid ROM component can control all of the devices given in section 1.2 Controlled Devices, the portion of this component that controls wireless communications is special-purpose. This component is an extension of the `ichneumon` component for use with hybrid ROMs. Because it inherits the `ichneumon` component characteristics described in section 2.2, when this component is used together with wireless communications, the VRAM allocation status must be changed using the WVR library. As with `mongoose.TWL`, when this component is running on NITRO hardware, it is limited to controlling the five devices given in sections 1.2.1 through 1.2.5

2.5 racoon.TWL

This component (the default for limited ROMs) can also control all of the devices given in section 1.2 Controlled Devices. Dedicated for use with TWL hardware only, this component cannot run on NITRO hardware.

2.6 ferret.TWL

This special component for limited ROM builds can control all of the devices given in section 1.2 Controlled Devices except wireless-communication features and cameras. Dedicated for use with TWL hardware only, this component cannot run on NITRO hardware.

Because this component does not include wireless-communication or camera functionality, its size is smaller than the `racoon.TWL` component.

3 Selecting a Component

The component to be used is specified to the `makerom` tool that gathers the ARM9-side executable image (the application) and the ARM7-side executable image (the component) into a single binary file.

Specifically, this specification is made in the `Arm7` section of the `RSF` file provided to `makerom.exe` or `makerom.TWL.exe`. For details, see the description of `makerom` in the *Function Reference Manual*. When using an `RSF` file in its default state with a command-line build environment, select the component to be used by explicitly specifying the component type in the `DEFAULT_COMP_ARM7` variable before the `commondefs` reference in the application's makefile. If nothing is specified, the `mongoose` component is set as default for NITRO ROMs, the `mongoose.TWL` component is set as default for hybrid ROMs, and the `racoon.TWL` component is set as default for limited ROMs.

4 Identifying a Component

This section gives the method used by an application at run time to identify whether `ichneumon` or `ichneumon.TWL` is the currently running component.

1. Call `PXI_IsArm7CallbackReady(PXI_FIFO_TAG_WVR)` after calling `PXI_Init` . If `TRUE` is returned, the component is `ichneumon`.
2. If `FALSE` is returned in step 1, the component is another type.

5 Precautions

To run a component, there must be a memory space sufficient for holding the program itself. Although TWL hardware includes an ARM7-dedicated work RAM, this work RAM alone is not enough to hold a program. With the TWL-SDK, programs are positioned by allocating the shared ARM9 and ARM7 work RAM (16 KB x 2) to the ARM7. Take care not to reallocate this shared work RAM back to the ARM9.

With wireless-communication control programs, allocating this work RAM is still not sufficient. Under the TWL-SDK, the memory space allocated as the system-reserved area of main memory is also used to hold ARM7-side programs. Take care that this system-reserved area (ranging from `0x027e0000` to `0x02800000` in NITRO mode and from `0x02F80000` to `0x02FE40000` in TWL mode) is not used for other purposes. With the TWL-SDK, by default the data TCM is positioned in ARM9 memory at address `0x027e0000` when using NITRO mode, and at address `0x02FE0000` when using TWL mode.