Technologic SYSTEMS
www.embeddedarm.com (480) 837-5200

Getting started with TS-7KV for TS-7000

Hardware Overview

The TS-7KV is a multi-function PC104 daugther board that provides Video for TS-7000 ARM and other features like:

- 16-bit color/640X480 video resolution
- 8MB dedicated video memory running @ 95Mhz.
- Simple and fast video accelerator
- Accelerated Linux framebuffer driver available
- Standard DB15 VGA connector or 10 pin header
- Forward-compatible through adapter boards
- All features implemented on FPGA ensures long-term availability (no obsolete or proprietary graphics chips)
- 24 buffered, 5V tolerant GPIO lines (16 output, 8 input)
- 5 megabaud serial port
- Optional RS485, half or full, with automatic half-duplex transmitter enable/disable
- Optional 8 channel 200ksps 16-bit ADC
- Optional SJA1000 compatible CAN controller

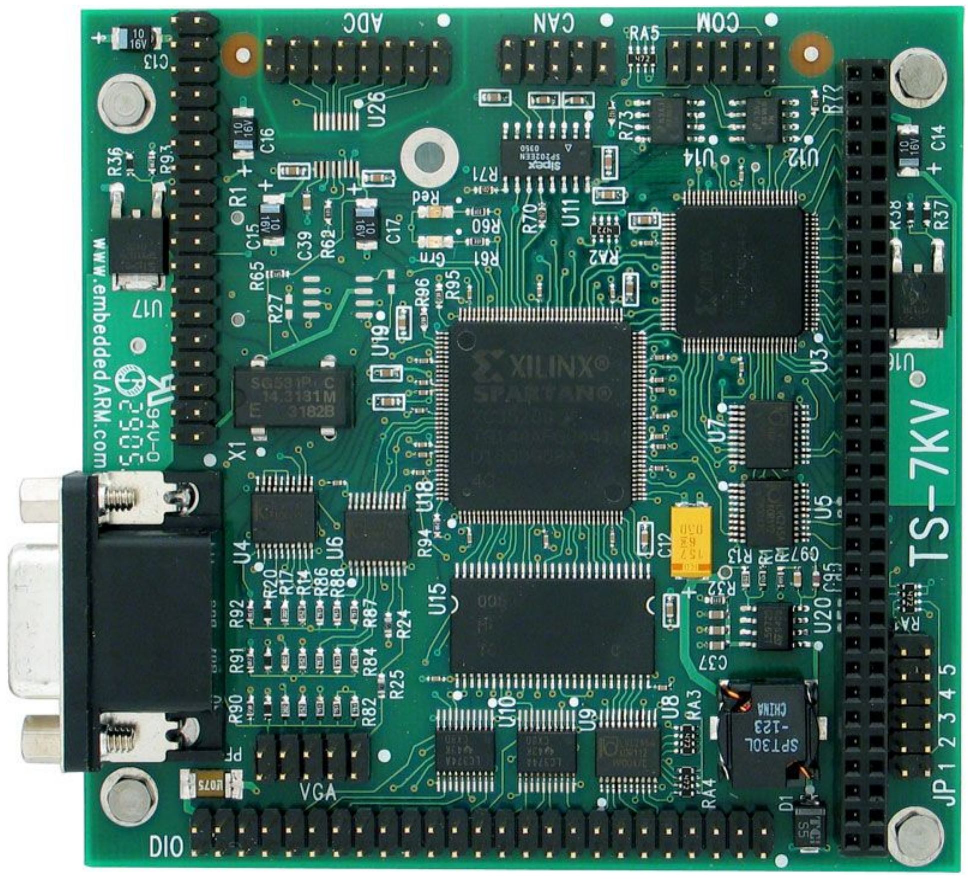

Table 1: Jumper settings for Base/FPGA/Video address selection

I/O Adrress	JP1	JP2	FPGA 32-bytes I/O	Video I/O
E0 - E7	**OFF**	**OFF**	00 - 1F	000000 - 0FFFFF
E8 - EF	**ON**	**OFF**	20 - 3F	100000 - 1FFFFF
F0 - F7	**OFF**	**ON**	40 - 5F	200000 - 2FFFFF
F8 - FF	**ON**	**ON**	60 - 7F	300000 - 3FFFFF

Table 2: Jumper settings for IRQ

IRQ	JP4	JP5
None	OFF	OFF
IRQ6	ON	OFF
IRQ7	OFF	ON
	ON	ON

Table 3: Base Register Map: TS-7KV

I/O Address	Description	Data	Bits and such
BASE + 0	Board identifier #1	Read only	Fixed value: 0x41 hex
BASE + 1	Board identifier #2	Read only	Fixed value: 0x20 hex
BASE + 2	PLD version register	Read only	Fixed value
BASE + 3	Reserved		
BASE + 4	Control register 0	R/W	**Bit 7:** Done bit; FPGA firmware loaded. **Bits 2-6:** reserved **Bit 1:** FPGA config, set at reset **Bit 0:** FPGA ready, cleared at reset
BASE + 5	Control register 1	R/W	**Bit 7:** HSWAP; Control bit that drives the FPGA (1 = no PU) (0 = pull-up resistors enabled) **Bit 6:** INIT; Control bit that drives the FPGA (1 = true). INIT is open-drain and has pull-up resistor. Reads return value at FPGA pin. **Bit 5:** RDWR; Control bit that drives the FPGA (1 = ready). It must be either high to read back configuration or low to write config data. **Bit 4:** CS; Control bit that drives the FPGA (1 = true) **Bit 3:** PROG; Control bit that drives the FPGA (1 = true) **Bit 2:** M2; mode control bit that drives the FPGA **Bit 1:** M1; mode control bit that drives the FPGA **Bit 0:** M0; mode control bit that drives the FPGA
BASE + 6	FPGA configuration write data	Write only	Write cycles to this location writes configuration data into the FPGA.
BASE + 7	Jumpers and options	Read only	**Bit 7:** ADC option **Bit 6:** CAN option **Bit 5:** Jumper 5 (1=on, 0=off) address decode **Bit 4:** Jumper 4 (1=on, 0=off) address decode **Bit 3:** Jumper 3 (1=on, 0=off) reserved **Bit 2:** Jumper 2 (1=on, 0=off) interrupt selection **Bit 1:** Jumper 1 (1=on, 0=off) interrupt selection **Bit 0:** reserved

USING THE WINDOW REGISTER

The communication with the TS7KV FPGA, and thus with all the sub-devices implemented on it, is provided through a 32-byte I/O space selected as described in previous table 1. The 32-byte I/O space accesses 1 of 5 sub-devices depending on the value loaded into the 16-bit SWIN register located at BASE+1E. The remaining 30 bytes, from BASE+00 to BASE+1D, are a window into the register space of the selected sub-device. If the selected core on the FPGA needs more than 30 bytes of register space, it is possible to switch the window in steps of 16-bytes by writing to bits 3-0 of the SWIN register. The following table describes this register:

Table 4: 16-bit SWIN register at BASE+1E

I/O Address	Description	Data	Bits and such
BASE + 1E	SWIN register high	R/W	**Bits 15-8**: reserved **Bits 7-4**: Slave device select: set **0000** for 16550 RS232/RS485 serial UART; set **0001** for SJA1000 CANbus controller. set **0010** for LTC1867 SPI ADC. set **0011** for digital GPIO line registers. set **0100** for red/green on board LED control. set **0101** for video control I/O block. **Bits 3-0**: slave window offset in 16-byte units, for devices that requires more than 30 bytes of address space.

USING THE FPGA VIDEO CORE: TS-VIDCORE

After selecting the video sub-device by writing the value 5 to bits 7-4 of the SWIN register, the video control registers appear at the base of the 32-byte I/O adrres space, as described previously. All the functionality of the TS-VIDCORE is controlled through only five 16-bit registers, thus occupying 10 bytes of the 30-byte TS-7KV address space for sub-devices. The next table describes the video control registers.

Table 5: 16-bit registers map for Video control

I/O Address	Description	Data	Bits and such
BASE + 0	BLTCTRL: Bit blit control register	R/W	**Bits 15-13:** upper 3 bits of box pixel width **Bit 12:** bit blit source mode (0 – rectangle, 1 – linear) **Bits 11-6:** upper 6 bits of destination address of bit blit operation **Bits 5-0:** upper 6 bits of start address of bit blit operation
BASE + 2	BLTSZ: Bit blit width and height register	R/W	**Bits 15-9:** box pixel width (lower 7 bits) **Bits 8-0:** box pixel height (0-512)
BASE + 4	SRCBLT: Bit blit source	R/W	**Bits 15-0:** lower 16 bits of source address or pixel fill color
BASE + 6	DSTBLT: Bit blit destination	R/W	**Bits 15-0:** lower 16 bits of destination address
BASE + 8	VIDCTRL: Video control register	R/W	**Bit 11:** raster page committed (Read Only) **Bit 10:** bit blit operation in progress (Read Only) **Bit 9:** horizontal sync enabled **Bit 8:** vertical sync enabled **Bit 7:** bit blit direction **0** – top to bottom: SRCBLT and DSTBLT are top-left corner addr **1** – bottom to top: SRCBLT and DSTBLT are bottom-left corner addr **Bit 6:** pixel fill enable (SRCBLT is pixel color instead of addr) **Bits 5-3:** raster page select (0-7) – selects screen being displayed **Bits 2-0:** bus page select (0-7) – selects screen accessible via PC104 memory space

NOTES ON BIT BLIT OPERATION

The bit blit operation begins on write of DSTBLT register. If the DSTBLT is written again before bit blit operation completes, the FPGA bus (wishbone) cycle is stalled until previous operation completion. The bit blitter clones all bit blit registers on start of bit blit operation such that new values can be loaded in preparation for next bit blit.
There is a demo application at ftp://ftp.embeddedarm.com/ts-7kv/bitblt-demo that executes a bit blit operation using TS-7KV video core. It moves Technologic Systems' logo arround the screen 2000 times per second.

TS-VIDCORE FRAMEBUFFER DRIVER FOR LINUX

Technologic Systems provides a framebuffer driver that manages the TS-VIDCORE. The video registers are properly handled inside the driver so other Linux Kernel layers can interact with the TS-VIDCORE using the Framebuffer Device API. The driver enables video interaction from user-space using the /dev/fb framebuffer entry.
The Linux Framebuffer driver is integrated in the new official kernel release for TS7200, named **ts9**. Default kernels shipped with TS-7000 do not have video support, so you must update your kernel image from redboot in order to support TS-7KV. New kernels for TS-7200/TS-7250, new kernel modules and other related and necessary files are available in the directory ts-7kv of Technologic Systems' FTP server (ftp://ftp.embeddedarm.com/ts-7kv/).

If you **do not have the ts9** kernel release, use the next sections to guide you on how to update your TS7000 system for running all the TS-7KV functionalities.

INSTALLING THE KERNEL AND MODULES

There are many ways to get the new kernel into RedBoot. One way involves using the internet. To continue, make sure you have configured RedBoot with a network configuration that can reach the internet. You may use the RedBoot "fconfig" command to set network parameters. Get to the RedBoot prompt by hitting Ctrl-C key immediately after power-on and type the following commands:

```
RedBoot> load -v -r -b 0x00218000 -m http -h 67.40.67.44 /ftp/ts-7kv/vmlinux-ts7200-ts9.bin
-OR- (if you have a TS-7250):
RedBoot> load -v -r -b 0x00218000 -m http -h 67.40.67.44 /ftp/ts-7kv/vmlinux-ts7250-ts9.bin

RedBoot> fis delete vmlinux
RedBoot> fis create -b 0x00218000 -l 0x160000 vmlinux
```

If RedBoot gives you an error about not understanding the "-m http" option, you have an older version of RedBoot and must instead load the kernel via a local TFTP server, for example:

```
RedBoot> load -r -b 0x00218000 -h 192.168.0.1 /tftp/ts-7kv/vmlinux-ts7250-ts9.bin
```

Once the new kernel is installed and booted, you need to extract the new modules for TS-7KV framebuffer and serial port into /lib/modules. To do this, you must upload the file linux24-ts9-modules.tar.gz to the TS-72XX using the embedded FTP server and extract using the "tar" command into the / directory:

```
> tar zxvf  linux24-ts9-modules.tar.gz -C /
```

Once the new modules are extracted, you may optionally delete the old module directory for the old kernel in /lib/modules/ to reclaim flash space.

INSTALLING THE CONSOLE KIT

For you to see a text-mode console login prompt when you boot your TS-7200/TS-7250, you must extract the file ts7kv-tslinux-console-kit.tar.gz also into the / directory.

```
> tar zxvf  ts7kv-tslinux-console-kit.tar.gz -C /
```

This file contains an init.d startup script that detects the TS-7KV, loads the FPGA firmware, loads the TS-7KV modules (and USB keyboard and mouse modules also for convenience), and starts a login prompt (getty) on the newly formed Linux text console. Note that this tarfile is only good for the onboard flash mini-Linux installation. To use the TS-7KV on a 256Mb CF development kit Debian version, you will need, instead, to do:

```
> tar zxvf  ts7kv-debian-console-kit.tar.gz -C /
```

After installing kernel, module and console kit for TS-7KV, you should be able to use a Linux terminal after conecting a display on TS-7KV's video output, conecting USB mouse and keyboard, and rebooting the system.

INSTALLING THE QT/EMBEDDED GRAPHICAL LIBRARY

It is possible to builld advanced graphical user interface applications on top of TS-7KV for ARM-Linux embedded systems by using the QT/Embedded libraries. The development using Qt/Embedded is based on C/C++ Linux tools. Technologic Systems does provide, at the same ftp directory, a compiled package of the free and open source version of Qt/Embedded 3.3.4. The ts-7kv-qtembedded-full.tar.gz contains the entire Qt/Embedded compiled library as well as examples. There is also a small footprint Qt/Embedded version named ts-7kv-qtembedded-compact.tar.gz, in order to save flash space. To install the graphical library, extract the file into the / directory:

```
> tar zxvf  ts-7kv-qtembedded-compact.tar.gz -C /
```

It will create and fill the /usr/local/qt-embedded-free-3.3.4/ directory. The Qt/Embedded examples are located into directory /usr/local/qt-embedded-free-3.3.4/examples/. To run the hello world example you can execute the hello.run script. It defines the Qt/Embeded environment variables; creates symbolic links to the framebuffer, mouse and keyboard devices into the /dev directory; and calls the hello application as a server.

```
> cd usr/local/qt-embedded-free-3.3.4/examples/
> ./hello.run
```

To learn how to program GUI application using the Qt/Embedded API, please refer to the specific documentation:

- http://doc.trolltech.com/3.3/index.html

ADDITIONAL FEATURE: USING THE TS-7KV CAN CONTROLLER

The CAN controller implemented inside TS-7KV FPGA is Philips SJA1000 compatible. To make the SJA1000 128-byte address space appears on base address, you need to write the value 1 to bits 7-4 of the SWIN register, and then select the wanted register page using bits 3-0 of the SWIN register.
The linux driver for TS-CAN1, another PC/104 daughter board using SJA1000 provided by Technologic System, also supports the TS-7KV. All the information provided by TS-CAN1 documentation also applys for TS-7KV boards.

For further information on how to install and use the CAN driver for Linux, find the Getting Started with TS-CAN1 manual at http://www.embeddedarm.com/Manuals/gs_can1.pdf

ADDITIONAL FEATURE: USING THE TS-7KV SERIAL PORT

The TS-7KV implements a 16550 RS232/RS485 serial UART module. To use this subdevice it is necessary to write the value 0x00 to the SWIN register at adrress BASE+1E. This will make the serial registers appear at the 30-byte window adress space for subdevices, starting at BASE+00, where BASE is configured as shown in table 1.
You may research the internet for further information about the very common 16550 UART specification, such as registers map. For example:

- http://www.freebsd.org/doc/en_US.ISO8859-1/articles/serial-uart/

If you have properly installed the console kit, the TS-7KV serial port will be detected during boot time, and then the Linux driver, named **ts7kvserial**, will be loaded by default, enabling you to use the /dev file system entry with Linux systen calls (open, read, write, close) from user space.

ADDITIONAL FEATURE: USING THE TS-7KV GPIO

The TS-7KV implements a digital General Purpose I/O module. To use this subdevice it is necessary to write the value 0x30 to the SWIN register at adrress BASE+1E. This will make the GPIO registers appear at the 30-byte window adress space for subdevices, starting at BASE+00, where BASE is configured as shown in table 1.

There are two 16-bit registers for handling the GPIO lines. The first, at BASE+00, controls the 16 output lines, while the 8 lower bits of the second register, at BASE+02, control the 8 input lines. Outputs are 3.3V and can sink/source 24mA. Inputs are 5V tolerant, 3.3V w/CMOS thresholds.

ADDITIONAL FEATURE: USING THE TS-7KV ADC

The TS-7KV implements a LTC1867 SPI ADC module. To use this subdevice it is necessary to write the value 0x20 to the SWIN register at adrress BASE+1E. This will make the ADC registers appear at the 30-byte window adress space for subdevices, starting at BASE+00, where BASE is configured as shown in table 1.

The ADC sub-device is implemented using one single 16-bit register at BASE+00. Writes to this registers send the 7-bit command word. Conversions are always taking place at 190 kbps and reads give back the last data converted by the ADC. For further information, refer to the LTC1867 documentation:

- http://www.linear.com/pc/productDetail.do?navId=H0,C1,C1155,C1001,C1158,P2497&action=viewall

SUPPORT

Email: support@embeddedarm.com

DOCUMENT HISTORY
==================
10.26.2005 – Created