OVERVIEW

TS-7KV is a PC/104 (standard format) peripheral board that works with all Technologic Systems ARM embedded computers to provide VGA video function. The TS-VIDCORE is implemented in the on-board Xilinx Spartan-3 **FPGA**, which is re-configurable by software through the Linux OS during runtime. Hardware features and specifications include:

- ✔ 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

The **TS-7KV** multi-function peripheral board provides additional features which are also included in the default TS-7KV FPGA bitstream provided by Technologic Systems:

- ✔ 24 buffered, 5V tolerant GPIO lines (16 output, 8 input)
- ✔ 5 megabaud capable RS232 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

Notes

All features implemented on FPGA ensures long-term availability (no obsolete or proprietary graphics chips).

HARDWARE CONFIGURATION

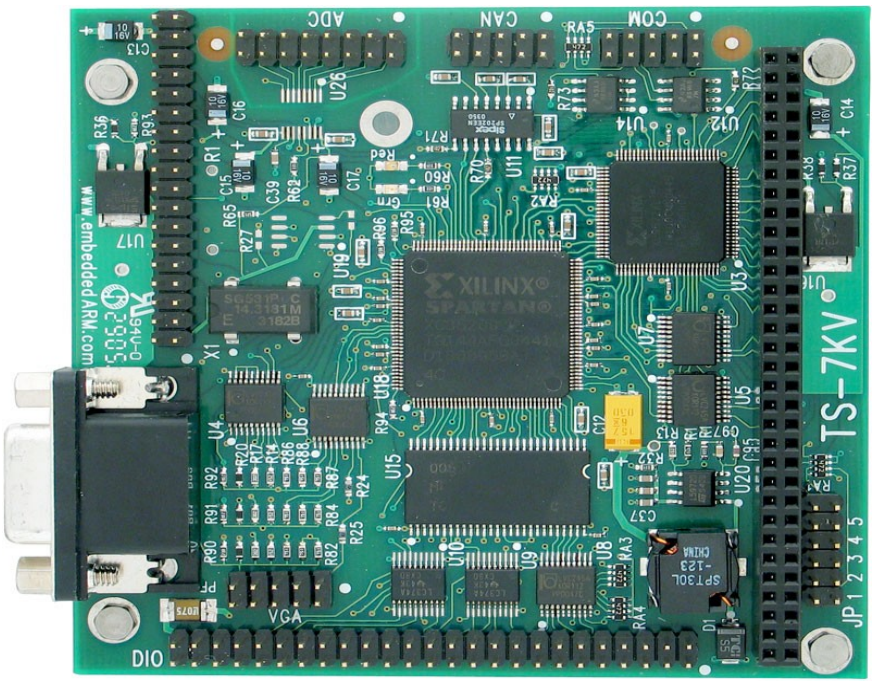

Table1: Jumper settings for I/O Addr

I/O Addr	JP1	JP2
0xE0 - 0xE7	OFF	OFF
0xE8 -0xEF	ON	OFF
0xF0 - 0xF7	OFF	ON
0xF8 - 0xFF	ON	ON

Table2: Jumpers settings for IRQ

IRQ	JP4	JP5
None	OFF	OFF
IRQ6	ON	OFF
IRQ7	OFF	ON
IRQ5	ON	ON

HARDWARE CONFIGURATION

Besides the basic TS-7KV I/O address (PLD registers), the jumpers #1 and #2 also select the FPGA and Video I/O memory spaces. The table below shows the configuration options:

Table3: Jumper settings for PLD, FPGA and Video I/O address spaces

JP1	JP2	PLD Registers	FPGA Registers	Video Memory
OFF	**OFF**	BASE+ (E0 to E7)	BASE+ (00 to 1F)	BASE+ (000000 to 0FFFFF)
ON	**OFF**	BASE+ (E8 to EF)	BASE+ (20 to 3F)	BASE+ (100000 to 1FFFFF)
OFF	**ON**	BASE+ (F0 to F7)	BASE+ (40 to 5F)	BASE+ (200000 to 2FFFFF)
ON	**ON**	BASE+ (F8 to FF)	BASE+ (60 to 7F)	BASE+ (300000 to 3FFFFF)
		To access the 8-bytes of I/O address space for the PLD 8-bit registers, use **BASE=0x11E0_0000**	To access the 32-bytes of I/O address space for the FPGA 16-bit registers, use **BASE=0x21E0_0000**	To get direct access to the Video / Framebuffer memory, use **BASE=0x2180_0000**

PLD REGISTERS

Table4: TS-7KV PLD register map

I/O Addr	Description	Data	Bits and such
Base + 0	Board ID #1	Read only	Fixed value: 0x41 hex
Base + 1	Board ID #2	Read only	Fixed value: 0x20 hex
Base + 2	PLD version reg	Read only	Fixed value
Base + 3	Reserved		
Base + 4	Control reg #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 reg #1	R/W	**Bit 7:** HSWAP; Control bit that drives the FPGA (1=no pull-up) (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 drives the FPGA (1=true) **Bit 3:** PROG; Control bit 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 config write data	Write only	Write cycles to this location writes config 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

Use PC/104 8-bit I/O cycles through memory base address **0x11E0_0000** in order to access the PLD 8-bit wide registers mapped in the 8-bytes of I/O space.

FPGA REGISTERS

The communication with the TS-7KV FPGA, and thus with all the sub-devices implemented on it, is provided through a 32-byte I/O memory space selected as described in the table 3. The FPGA 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 memory 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:

Table5: FPGA 16-bit SWIN register at Base+1E

I/O Addr	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 CAN bus 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.

Notes
Use PC/104 16-bit I/O cycles through memory base address **0x21E0_0000** in order to access the FPGA 16-bit wide registers mapped in the 32-bytes of I/O space. This applies to any sub-device included on the FPGA.

TS-VIDCORE : THE TS-7KV VIDEO CORE

After selecting the video sub-device by writing the value **0x5** to bits 7-4 of the SWIN register at **Base+1E**, the video control registers appear at the base of the FPGA I/O space. All the functionality of the TS-VIDCORE is controlled through only five 16-bit registers which occupy 10 bytes of the 30-byte FPGA I/O space for sub-devices. The next table describes the video control registers.

Table6: TS-VIDCORE register map

I/O Addr	Description	Data	Bits and such
Base + 0	BLTCTRL: Bit blit control reg	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 reg	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 dest	R/W	**Bits 15-0**: lower 16 bits of destination address

Table6: TS-VIDCORE register map

I/O Addr	Description	Data	Bits and such
Base + 8	VIDCTRL: Video control reg	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 that executes a bit blit operation using TS-7KV video core. It moves Technologic Systems' logo around the screen 2000 times per second.

- **ftp://ftp.embeddedarm.com/pc104-peripherals/ts-7kv/ts-72xx-linux24/bitblt-demo**

ADDITIONAL FEATURE: 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 **0x1** to bits 7-4 of the SWIN register at **Base+1E** (FPGA I/O Address), and then select the wanted register page using bits 3-0 of the same SWIN register.

The Linux driver for TS-CAN1, another PC/104 daughter board using SJA1000 provided by Technologic Systems, also supports the TS-7KV. All the information provided by TS-CAN1 documentation also applies for TS-7KV boards.

For further information on how to install and use the CAN driver for Linux with TS-7KV, find the Getting Started with TS-CAN1 manual at:

- **http://www.embeddedarm.com/documentation/ts-can1-manual.pdf**

The lincan driver binary for ARM can be found at:

- **ftp://ftp.embeddedarm.com/pc104-peripherals/ts-can1/ts-72xx-linuxr24/lincan-ts11.o**

To load this driver with TS-7KV, use the command insmod as:

```
$ insmod -f lincan-ts11.o hw=ts7kv
```

ADDITIONAL FEATURE: THE TS-7KV SERIAL PORT

The TS-7KV implements a 16550 RS232/RS485 serial UART module. To use this sub-

device it is necessary to write the value **0x00** to the SWIN register at **Base+1E**. This will make the serial registers appear at the 30-byte window address space for sub-devices, starting at Base+00 (FPGA I/O Address).

You may want to 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 system calls (open, read, write, close) from user space.

ADDITIONAL FEATURE: THE TS-7KV GPIO

The TS-7KV implements a digital General Purpose I/O module. In order to use this sub-device, it is necessary to write the value 0x30 to the SWIN register at address **Base+1E** (FPGA I/O Address). This will make the GPIO registers appear at the 30-byte window address space for sub-devices, starting at Base+00 (FPGA I/O Address).

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

ADDITIONAL FEATURE: THE TS-7KV ADC

The TS-7KV implements a LTC1867 SPI ADC module. To use this sub-device it is necessary to write the value 0x20 to the SWIN register at address **Base+1E** (FPGA I/O Address). This will make the ADC registers appear at the 30-byte window address space for sub-devices, starting at Base+00 (FPGA I/O Address).

The ADC sub-device is implemented using one single 16-bit register at Base+00 (FPGA I/O Address). 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

SOFTWARE SUPPORT

CONFIGURING THE FPGA USING LINUX

The Xilinx FPGA on the TS-7KV can be re-configurable during runtime by the Linux OS. Therefore, the user can either load another FPGA design provided by Technologic Systems or develop your custom design. Technologic Systems provides a Linux application utility that loads a bitstream into the TS-7KV FPGA. It can be found at:

- **ftp://ftp.embeddedarm.com/pc104-peripherals/ts-7kv/ts-72xx-linux24/load_ts7kv**

The default TS-7kV bitstream provided by Technologic Systems is available at:

- **ftp://ftp.embeddedarm.com/pc104-peripherals/ts-7kv/ts-72xx-linux24/ts7kv.bit**

To load the default bitstream, enter the command below at the TS-7000 Linux prompt:

```
$ load-ts7kv ts7kv.bit
```

FRAMEBUFFER DRIVER

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 TS-7KV Framebuffer video memory, as described in table 3, is located at base address **0x2180_0000**.

The Linux Framebuffer driver for TS-7KV is included in the official TS-Kernel releases, greater than **ts9** version. Older TS-Kernels shipped with TS-7000 computers do not have video support, so you must update your system (Kernel and File System) from Redboot in order to support the TS-7KV hardware. System updating instructions for TS-Kernel, version ts9, can be found at:

- **http://www.embeddedarm.com/products/board-detail.php?product=TS-7KV#**

Notes
If your TS-Kernel release is older than ts9 version, read the next section named ”The Console Kit” in order to guide you on how to update your TS-7000 file system.

FRAMEBUFFER EXAMPLES

Technologic Systems provides some examples and source code on how to program the TS-7KV Framebuffer with C from Linux user-space. These files are available at:

- **ftp://ftp.embeddedarm.com/ts-arm-sbc/ts-7200-linux/samples/arm-binaries/circle-7kv**
- **ftp://ftp.embeddedarm.com/ts-arm-sbc/ts-7200-linux/samples/arm-binaries/gradients-7kv**
- **ftp://ftp.embeddedarm.com/ts-arm-sbc/ts-7200-linux/binaries/ts-logo-splash**

For example, to view a Technologic System's logo on the display, enter:

```
$ cat ts-logo-splash > /dev/fb/0
```

THE CONSOLE KIT

For you to see a text-mode console login prompt when you boot your TS-7000 SBC with TS-7KV, you need to configure your system. Technologic Systems provides a tarball that 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. Find the tarball files for TS-Linux and Debian distributions, respectively, at:

- **ftp://ftp.embeddedarm.com/ts-arm-sbc/ts-7200-linux/binaries/ts7kv-tslinux-console-kit.tar.gz**

- **ftp://ftp.embeddedarm.com/ts-arm-sbc/ts-7200-linux/binaries/ts7kv-debian-console-kit.tar.gz**

You must extract one of these files into the root directory:

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

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

CURSOR BLINKING RATE

The blinking rate of the framebuffer cursor can be changed to any value as shown below:

```
$ echo NEW_BLINKING_RATE > /proc/sys/dev/fb/cursor_blink_rate
```

THE QT/EMBEDDED GRAPHICAL LIBRARY

It is possible to build 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 provides a compiled package of the free and open source version of Qt/Embedded 3.3.4:

- **ftp://ftp.embeddedarm.com/ts-arm-sbc/ts-7200-linux/binaries/qt-embedded-free-3.3.4-compiled-compact.tar.gz**
- **ftp://ftp.embeddedarm.com/ts-arm-sbc/ts-7200-linux/binaries/qt-embedded-free-3.3.4-compiled-full.tar.gz**

The second file contains the entire Qt/Embedded compiled library as well as examples, while the first one is a small footprint Qt/Embedded version which saves flash space. To install the graphical library, extract the selected file into the root 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**

CONTACT TECHNOLOGIC SYSTEMS

16525 East Laser Drive
Fountain Hills, AZ 85268
TEL 1.480.837.5200
FAX 1.480.837.5300

www.embeddedARM.com
support@embeddedARM.com

DOCUMENT HISTORY

Date of Issue/Revision	Revision Number	Comments
October 26, 2005	Preliminary DRAFT	PRELIMINARY release for first customer ship
August 15, 2006	1.0	Initial release
July 1, 2008	1.1	Fixed broken links, formatting
July 8, 2009	1.2	Updated Mailing Address