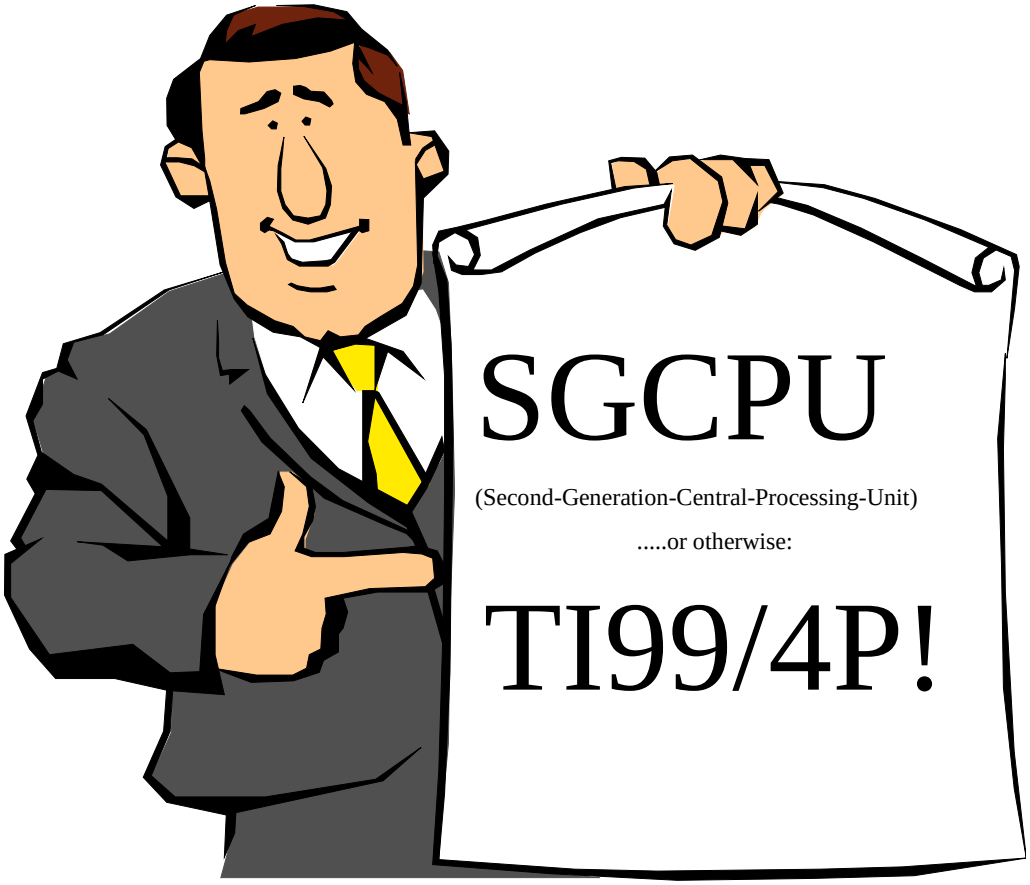

Hardware Description

Last Manual Edit: 2017-10-17
Translation by Bob Carmany

Actual versions at system-99 user-group

Table of Contents

Introduction

The SGCPU is a further extension of the existing TI99/4A. It is fully compatible and uses the same operating system. For copyright restrictions we have only as many SGCPU cards as the number of original TI99/4A motherboards that we have on hand (max. 50 pieces). They will be constructed with the original ROM. The name “Second Generation-CPU“ is because of the Bus definition of the Peri-Box. There are many unused signals that are designated “For Second Generation-CPU only“. TI had also planned to create an upgraded TI99/4A. That was never realized. If TI had used another CPU, we would not have the good old TMS9900 processor. First, we have an operating system and second they are able to be procured. We were itching to use the TMS99105, but they are difficult to find and not 100% compatible. We did not want to produce a second GENEVE. Eventually, our SGCPU along with its faster brother, the TI99/5P as a counterpart to the TI99/5 was created, of which I have an example. This has a TMS9995, I also have the operating system for it.. In any case, it is a CO-Pro-Card with a TMS320C25 signal processor and up to a 40MHz clock frequency. But that is for next year.

Description

The TI99/4P is an insertable card for the Peri-Box, but the “P“. Designates it as a TMS9900 at 12MHz. Standard configuration is with 32 KByte RAM on a 16 bit bus and 8 Kbyte of ROM. It does not have video or GPL which can be replaced by the EVPC and the HSGPL.The big innovation is the keyboard interface of the TI99/4P, a µC-Basis, which is a derivative of the 8031. This will run at up to 24MHz and takes over the entire coding. For a keyboard one can use a PC-keyboard with MF2 coding.This does not come with the SGCPU and must also be procured seperately. If you wish, we can naturally supply one for additional cost..

Configuration

The TI99/4P comes in the following sizes:

Version	RAM	AMS	XB-ROM	DSR	Price
Standard	32 Kbyte / 16 Bit	0 Kbyte	No	No	Ca. 280,- DM
Standard	32 Kbyte / 16 Bit	0 Kbyte	Yes 32 Kbyte	Yes 32 Kbyte	Ca. 300,- DM
Extended	AMS integrated	256 Kbyte / 16 Bit	Yes 32 Kbyte	Yes 32 Kbyte	Ca. 500,- DM
Extended	AMS integrated	1 Megabyte / 16 Bit	Yes 32 Kbyte	Yes 32 Kbyte	Ca. 600,- DM

Processor Kernel

The kernel of the TI99/4P is built with the TMS9900. It will run at 12MHz, however, for all of the clock-synchroned P-Box-cards the normal frequency of 3MHz is used. This is most important for the RS232. A 16MHz version is possible, but we will not undertake to build it. Speed can be gained by other means. But more later. The clock is set by the TIM9904A, that also takes over the RESET management. The required voltage is supplied by the three lines on board.

Memory

ROM

The operating system is 8Kbyes in size built on a 16-bit bus. There is also an 8Kbye DSR as well as a 16KByte ROM6 also on a 16-bit bus. The DSR area is for our own subprograms for the control of new hardware functions through SENILA,B etc. The 16KByte ROM6 is for the XBASIC interpreter in the XBASIC ROM. These ROMs are at this time the last “gadfly“ in the TI. What is contained in this ROM iis the very same interpreter that is in Extended Basic. Even with the HSGPL one cannot realize the full possible speed. The reason is that even though GROM access is 6 times faster, the interpreter only sees an 8-bit bus. The interpreter runs 90% of the time through the CPU in XB. By setting the CRU bits, ROM6 access is switched from external to internal in the 16-bit access. That shortens the run time of most programs by about 10-14%. It is really not possible to do more.

RAM

The RAM is the normal 32Kbytes that resides on a 16-bit bus. In the “Extended“ versions the 32K “Super-AMS“-compatibility is built in. This mapping technique is used in the ASGARD AMS board and also in the new American software system. The hardware is naturally not a copy (very understanbable!) but is a fully new development. This exported memory can be 256Kbyte or 1MB. There is also a seperate small PAD-RAM, that is permanently available (that is also always 1KByte). A few modules use this area for program errors (Minimem) that can be corrected thankfully with the HSGPL.

CRU

The CRU is just the same is in the TI99/4A. In the SGCPU the supported CRU bits in the TMS9901 that wait for a keypress so the new bit can be turned on are in the ROM6 16-bit buss and “Interrupt-level-sense“ directed by SENILA and SENILB. Also, the CRU-bit for the cassette control are physically there but the switching functionality is not. This might be added later.

Bus-Interface

The SGCPU recognizes all of the signals from the Peri-Box. The most important of these, Video-Interrupt, is externally stripped from the motherboard by the EVPC. Even so, the “Interrupt-level-sense“ direction is completely supported from the processor which uses all of the possible DMA-Protocols. It is also possible to have two processor cards like the TI99/4P and the TI99/5P in the Peri-Box. One can thus have intelligent Peripheral cards available. The high state of the control address lines AMA ,AMB and AMC can be seen with one’s own eyes. Basically, dummy cards like the HORIZON RAMDisk these are not decoded. A new edition of the HRD is not anticipated since most have one of the original versions..The Flex Cable Interface and the “RDBENA“ signal has been done away with and is no longer necessary. The light display of the BWG is bi-color. When reading data it is green. When the CPU receives a HOLD-Request, the LED is red. Reading access on an 8-bit bus both lights are on and the mixture produces yellow. The greener the light the faster the access.

Decoder

The decoder in this card is in a MACH chip, this one of the MACH2 family. It undertakes all of the decoding of the storage, CRU control and other functions of the TMS9901 and the mapping of the AMS emulation. In other words, it is programmed like its brother, the HSGPL. It even controls the expansion ports.

Expansion Port

This port is a 10-pin plug for a ribbon cable connection. There are eight low data bits, the write signal for 16-bit access and a 16-bit enable. This enable is the open signal that allows one to turn on internal storage and communicate directly with the CPU. If this is active, the low eight bits will be read through the expansion port and the high eight bits read through a cable to the Peri-Box. A possibility, for example, is to realise this through the use a 16-bit input port. We built this into the card. For example, as a normal Bus-Interface for the Peri-Box and decoded it In this case, in the last address in ROM0, where there is no code. This decoder signal allows one through the ENABLE16 signal to create a quasi port in the empty part of ROM0. Naturally, both cards must be connected by a ribbon cable. The mechanics must be done cleanly, with the Expansion-Port from the SGCPU connected to the top edge of the board just under the top

cover of the Peri-Box. A similar principle is used by PCs to differentiate between 8 and 16-bit cards. The signal ENABLE16 is really an open-collector-Signal, that is used in place of the old “RDBENA“. Other cards can also use this function. Ingenious?

Joystick and Keyboard

These two Interfaces were built to be fully compatible with the old TI99/4A. Both of the Joystick-Ports were realised and are connected with a 25-pin D-plug. Two 9-pin would not have been enough, and it was decided to include everything for the eventual Keyboard-Interface for the original TI-Keyboard signals as well as the signals for the planned Cassette-Port in this plug. This last was the special wish of the folks from Wiesbaden, who finally won with this “Hamster cage“.

The keyboard is normally connected with a DIN-plug and should be MF2-compatible. This same Interface has integrated in it a µC of the type 89C2051 allegedly with 20 Pins! This µC with an 8031-core has been built with a 2K-FLASH-EPROM and 128 bytes of RAM. It runs at up to 24MHz and is fastest 8031 in the world. The software is completely interrupt-driven, in asynchronous mode. After observing the first Meier Interfaces and having learned from them, we have avoided the technical problems that they experienced. A this time, only the MF2-keyboards with 11-bit-AT-code is supported, other versions will be completed as needed.

AEMS Emulation

In the extended versons of the SGCPU the AEMS (Asgard Expanded Memory System) is fully emulated. With only one small difference: The fuction here is more secure than my original AMS (borrowed from W. Winkler) which always failed over time. The function: In the AEMS (or its predessor the AMS) the normal 32K address area (both low-mem and high-mem) are not polled.. On the CPU-Address bus the highest four lines (A0 to A3) are not used for memory but are used for addressing the 16 mapping registers. These are the 8-bit Write-Read registers that can be addressed in the AEMS at CRU >1E00. As soon as the map-bit at CRU >1E02 is set, the contents of the registers are put into high memory. By filling the highest address with zero, the next four addresses mentioned previously are now accessible through A0 to A3. By mapping, one has the ability to not to end up with four fewer addresses but with an 8-bit wide register to access 8 more addresses. Also in these same four addresses one can have a 16 time increase in address area. For the TI, this is accomplished with a 32K-Memory Expansion of 512K words which allows 1MByte to be addressed. One needs suitable software because there needs to be proper control. If anyone needs more information, they should take a look at the data sheet for the 74LS612 memory mapper. This is so terribly expensive to do that I have simply programmed it into the MACH chip (in these cases, I often use this line of reasoning). The thing is that it is also as big as a TMS9901 (PSI) and caused many juicy errors. Programs for the AEMS V1.2 all run on the SGCPU. They are at times strained but still come through in the right state. They are always mapped in 4K blocks in the 32K with 8-bit wide per register so that you have 256*4K blocks, or 1024KB or 1MB.

Technical Data

Memory map

Address	Function	Size	Bus	AMS
>FFFF to >A000	Memory Expansion	24KB	16 bit	Yes
>83FF to >8000	Scratch-PAD	1K	16 bit	No
>7FFF to >6000	ROM6	2* 8K banked	16 bit	No
>5FFF to >4000	DSR	8K	16 bit	No
>3FFF to >2000	Memory Expansion	8K	16 bit	Yes
>1FFF to >0000	ROM0	8K	16 bit	No

ROM map

Address in EPROM-Pair	Address in 27C256	Function
>FFFF to >E000	>7FFF only 4K there >7000 two EPROMs	ROM6 EX-BASIC Bank 6002
>7FFF to >6000	>3FFF >3000	ROM6 EX-BASIC Bank 6000
>DFFF to >C000	>6FFF >6000	ROM4 for internal DSR
>5FFF to >4000	>2FFF >2000	ROM0 Operating System

Explanation of the Storage Areas

The addresses are a bit unusual with the second highest is +5V so that a smaller 24-pin EPROM can be used for the standard version. The EPROM pair sees >0000 in ROM0 as the CPU address and for example ROM0 as >0000 instead of ROM0 as the address >4000, everything clear? When one individually programs the EPROM one must halve the addresses since one will have the straight addresses and the other other EPROM will have only the odd addresses. The addresses are shifted from the CPU and the artifically-produced address A15 is not even connected. This is necessary for 8-bit access to the Peri-Box. Those who study the connection plan will find that the CPU has only 15 address lines, A0 to A14. They are actually only for 16-bit access. The 8-bit story is that it was built in by the developers externally, designed so that writing a two-byte word caused the CPU to stop and wait for a two-byte write pulse. Also when reading the CPU is also stopped while it receives a two-byte read pulse. I have naturally integrated this in a MACH chip in the SGCPU. A rather complicated circuit with an 8-but latch that cost a lot of time but was necessary to enable the red LED to be seen. It must , however, be in the Peri-Box.

CRU-map

Address	Function	..when set:
>1E02 >1E00	**MM** **AEMS-Register**	**Map-mode active** **16 Registers in ROM4**
>0F06 >0F04 >0F02 >0F00	**SENILB** **SENILA** **ROM6/16 bit** **internal DSR (ROM4)**	**Int.-level-sense B** **Int.-level-sense A** **ROM6 internal** **DSR-enable**
>03FF to >0000	**PSI (9901)** **Like the TI99/4A**	**PSI (9901)** **Like the TI99/4A**

AEMS map

Address	Function (even byte, direct Address!)
>401E	**AEMS-Register 15**
>401C	**AEMS-Register 14**
>401A	**AEMS-Register 13**
>4018	**AEMS-Register 12**
>4016	**AEMS-Register 11**
>4014	**AEMS-Register 10**
>4012	**AEMS-Register 9**
>4010	**AEMS-Register 8**
>400E	**AEMS-Register 7**
>400C	**AEMS-Register 6**
>400A	**AEMS-Register 5**
>4008	**AEMS-Register 4**
>4006	**AEMS-Register 3**
>4004	**AEMS-Register 2**
>4002	**AEMS-Register 1**
>4000	**AEMS-Register 0**

Pin layout of the 25-Pin SUB-D Connector

Signal Name	Pin on D-Sub 25 - Conn	Pin on TI-Keybrd
INT5	3	1
INT6	16	2
INT8	17	3
INT4	15	4
INT3	2	5
P5	9	6
INT7	4	7
SEL5	10	8
SEL4	23	9
INT9	5	10
INT10	18	11
SEL0	13	12
SEL1	12	13
SEL2	24	14
SEL3	11	15
VCC	1	
Joystick 2	6	
Motor CS1	7	
Audio Gate	8	
GND	14	
Joystick 1	19	
P11	20	
CS1 Output	21	
SPC	22	
-5 Volt	25	

Technical Data of the Configurations

Build Version	DSR	ROM	RAM	AMS	XB	M8	INT	DMA	EN16
Standard small	No	8K	32K	No	No	No	Yes	Yes	Yes
Standard large	Yes	64K	32K	No	Yes	Yes	Yes	Yes	Yes
Extended small	Yes	64K	256K	Yes	Yes	Yes	Yes	Yes	Yes
Extended large	Yes	64K	1M	Yes	Yes	Yes	Yes	Yes	Yes