TLL SILC-6219 Embedded Systems Design Module

User Manual
Version 3.0

Copyright Notice

Disclaimer

Information in this document is subject to change without notice and does not represent a commitment on the part of TLL.

TLL provides this document as is, without warranty of any kind, expressed or implied, including, but not limited to, the particular purpose. TLL may make improvements and/or changes in this manual or in the products(s) and/or the program(s) described in this manual at any time.

Information in this manual is intended to be accurate and reliable. However, TLL assumes no responsibility for its use, or for any infringements of rights of other parties, which may result from its use.

This document could include technical or typographical errors. Changes are periodically made to the information herein; these changes may be incorporated in new editions of the publication. This manual is provided solely and exclusively for educational use and this information or related products should not be used nor relied upon for any purpose except for education and training.

Technical Support

Please contact your local TLL authorized product representative for questions regarding hardware, software or applications issues. Any updates or patches will be sent to you automatically as long as your registration is current. The TLL products are designed to be supported remotely by allowing viewing of the user’s desktop. It is highly recommended that the PC from which you are using TLL products is connected to an Internet link that allows Web browser access. In this way our technical support staff can view your desktop and work with you to understand and solve technical issues.

Table of Contents

Chapter 1- TLL SILC-6219

1.0 Introduction

The TLL SILC-6219 board is populated with an i.MX21 processor from Freescale. The i.MX21 utilizes the ARM926EJ-S core that is connected via the AMBA bus to broad range of peripherals. A number of these peripheral interfaces are accessible via standard connectors on the board. Key features of the board include:

- Ethernet, USB and RS-232 interfaces
- Standard 40 pin header with 36 GPIO pins.
- Two 90 pin mezzanine connectors for connection to SILC-5000 base Digital Systems Design Module.
- Standard 40 pin LCD interface
- JTAG pins for flash programming and CPLD programming
- Multiple boot-up options supported via user selectable jumpers
- Two user programmable LEDs and two user accessible jumpers for system configuration.
- uMon boot-loader for low-level real-time programming applications
- Linux version 2.6.16 with a preemptive kernel for high level programming applications

The features of the SILC-6219 provide a complete platform for active and systematic learning including fundamentals of boot loaders, Linux application programming and implementation of device drivers for interfacing to external hardware.

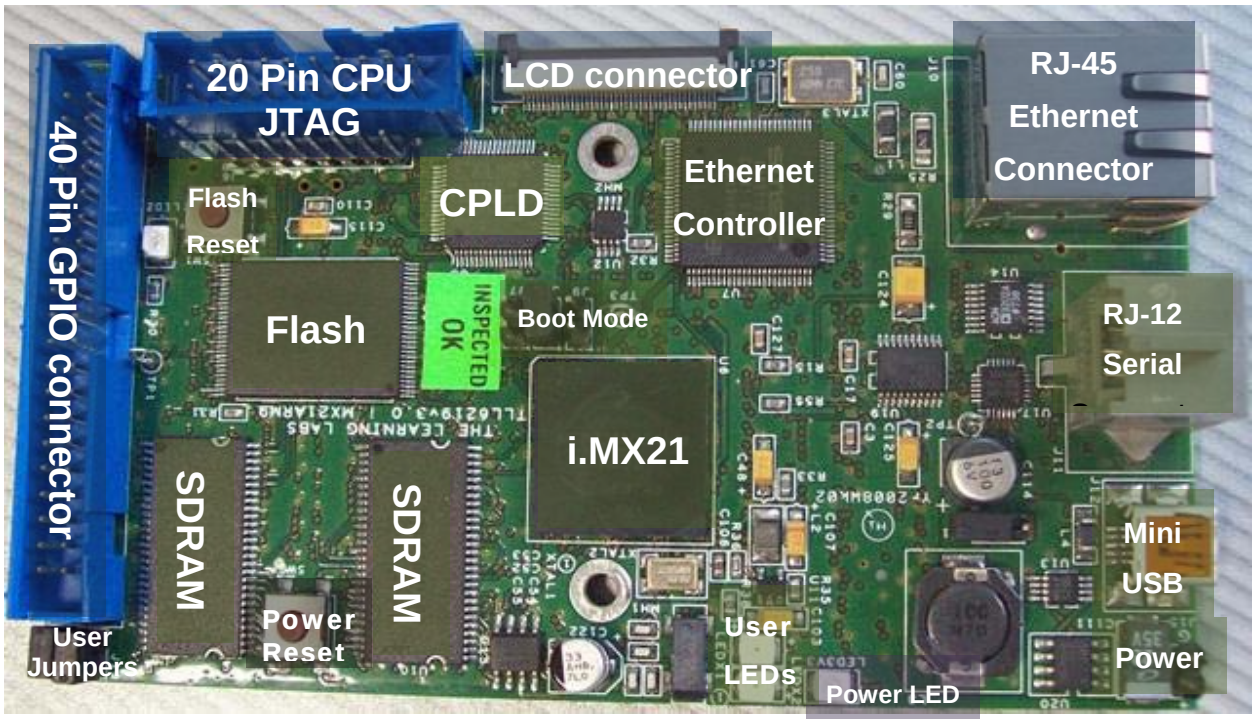
The TLL SILC-6219 Embedded System Design Module is shipped with the following components:

1. The SILC-6219 ARM based processor board
2. Interfacing Accessories
   - — Graphical LCD 320X240
   - — CMOS Camera – 1.3 Mega Pixels resolution
3. +18V/ 1A Power supply adapter
4. USB to Serial Converter with serial cable DB9 to RJ12
5. Ethernet Cross cable
6. Mini USB cable
7. Support CD
   - — TLL SILC-6219 Embedded systems development Tool chain Setup
   - — User Guide with sample programs
   - — Datasheets of various components in TLL SILC-6219
   - — Reference Guide for i.mx21 & MicroMonitor (uMON)

1.1 System Overview

The TLL SILC-6219 Module has following components:

- Microprocessor: Freescale i.MX21 System-on-Chip ARM9-based processor. The i.MX21 features full user transparency and access to the 32-bit address and data buses
- CPU core: ARM926EJ-S processor features a Jazelle technology enhanced 32-bit RISC CPU, 16KB of instruction & data cache.
- Memory: 64MB SDRAM & 16MB NOR Flash.
- Xilinx CPLD with 72 macro cells.
- High speed memory-mapped parallel interface: for communication with TLL SILC-5000 Digital Systems Design Module. This can also be used for expansion to high speed user applications with customized hardware that plugs into the mezannine connectors.
- Ethernet: with integrated PHY, support for 10Base-T and 100Base-TX.
- Mini USB OTG: with support up to full speed of 12Mbit/s.
- RS232: Serial communication support.
- LCD: smart LCD controller is mapped onto connector.
- CPU Jtag for debugging and CPLD Jtag for modifying the glue logic.
- 40 Pin GPIO port allows user to interface various peripherals to i.MX21, such as CMOS sensor interface, SPI, I2S, PWMs, Timers , Keyboard etc.



Chapter 2- Getting Started

2.1 Requirements

2.1.1 Host Computer System Requirements:

The following are the prerequisites for the host computer to run required software tools for the TLL SILC-6219.

- Pentium 1 GHz or higher -- Although lower speed processors will run the software tools, the desired results are not guaranteed.
- 512MB RAM – for average applications.
- Hard Disk Storage Space: Minimum 3-GB of space needed on hard disk.
- SVGA Monitor with the resolution of 1024x768 -- Monitor resolutions has been kept at 1024x768, which is ideal.
- Operating Systems: Windows 2000 (Service Pack-4), Windows XP (Service Pack-2)
- USB 2.0 ports – three ports are needed
- RS232 serial port
- Ethernet Network Interface Card (NIC): The network interface card available on PC has to be configured with a static IP address.

2.1.2 Software requirements:

The following software should be installed on the PC used for TLL SILC-6219 applications development.

- TLL SILC-6219 Embedded Systems Development Tool Suite.
- Cygwin (Version 1.5.24(0.156/4/2) or newer) – NOTE: Cygwin can have compatibility issues with other software so proceed very carefully. It is also recommended to install Cygwin only in the "c:\cygwin" directory
- Serial Communication Terminal program (HyperTerminal is default tool in Windows2000 and Windows-XP)

2.1.3 Hardware requirements:

The following hardware modules and accessories are required to start the experimentation

- TLL SILC-6219 Embedded Systems Design Module
- Wall 220/110 (50/60Hz) AC to DC Power Converter: 18V, 1 Amp
- USB – serial cable
- Mini USB cable

- RS232 serial cable (DB9 to RJ12)
- Cross Ethernet cable (Peer-to-Peer)
- TLL SILC-6219 CD

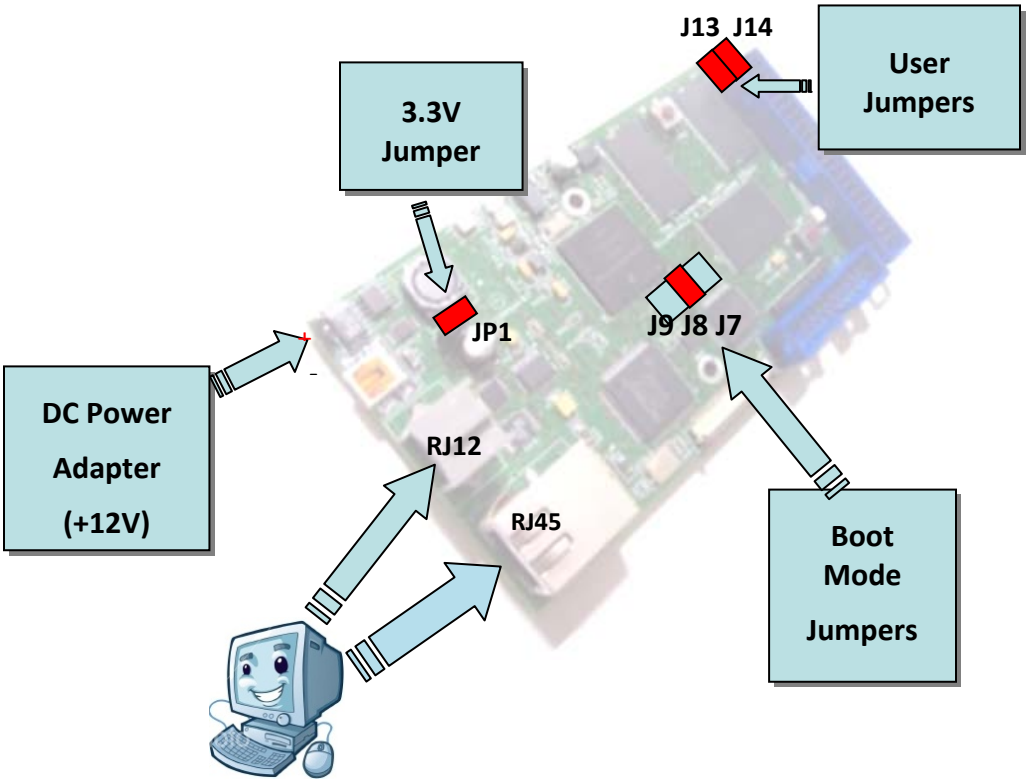
2.2 System Setup

2.2.1 Configuration & Cabling

The following steps need to be strictly followed for setting up testing TLL SILC-6219 Embedded System Design Module

1. Make sure jumper JP1 is closed.
2. Jumpers J13 and J14 should also be closed.
3. Boot mode Jumpers J8 should be closed and J9 & J7 should be open.
4. Power supply to be connected to the TLL SILC-6219 Module.
5. Connect the RS232 cable between Desktop PC & TLL SILC-6219 (Connect DB9 connector to PC and RJ12 connector to TLLSILC-6219)
6. Connect the Ethernet cross cable (Peer-to-Peer), between the desktop & TLL SILC-6219

Note: When using a Laptop as the Host PC it may be necessary to use a USB-Serial cable to connect to the RS232 port.



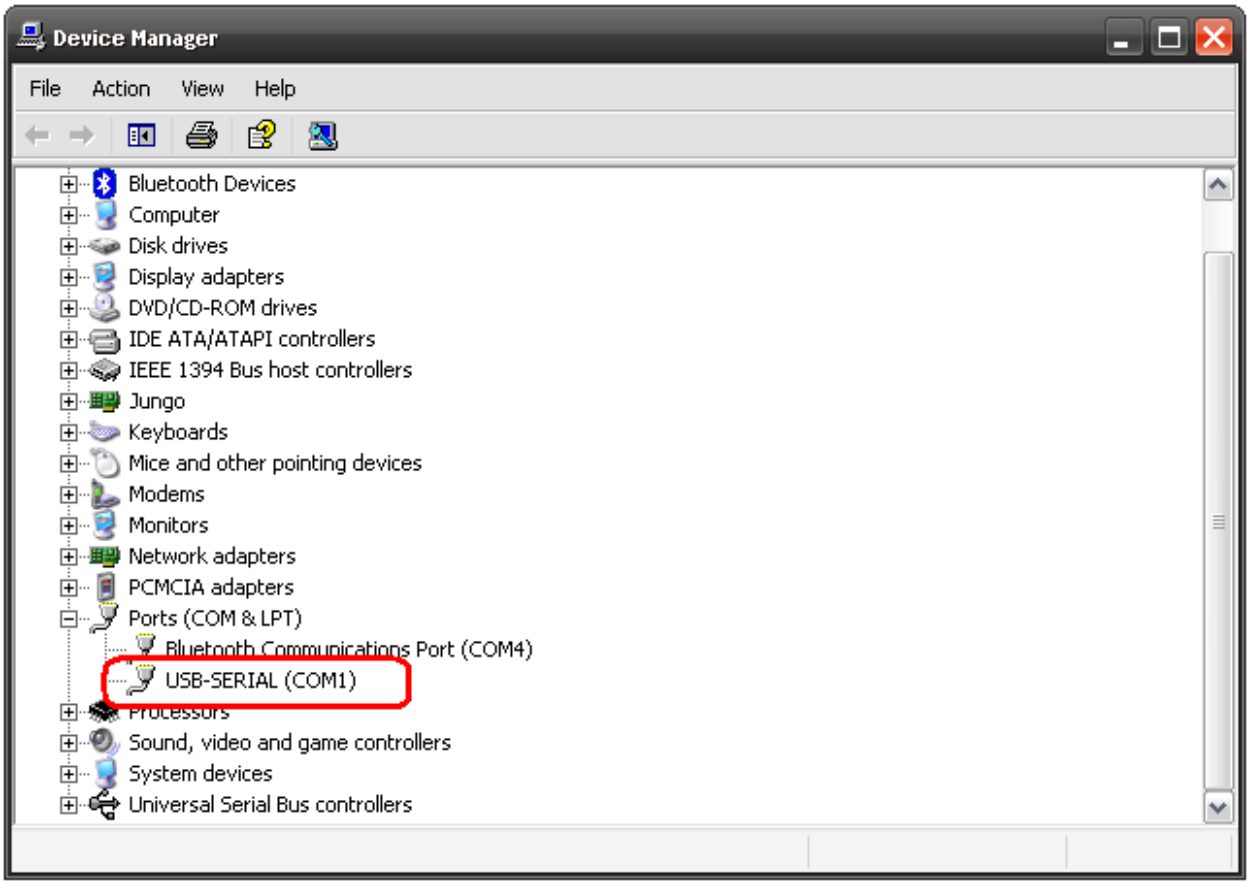
2.2.2 Monitor Boot-up

1. Open Serial Communication Terminal application (Hyperterminal/ booter (in Windows OS) or Minicom (in Linux OS).
2. Select the proper COM port and port parameter (HyperTerminal) as
   - Baud Rate—115200,
   - 8—bits,
   - None – Parity
   - Stop Bits -- 1
   - Hardware control – No.

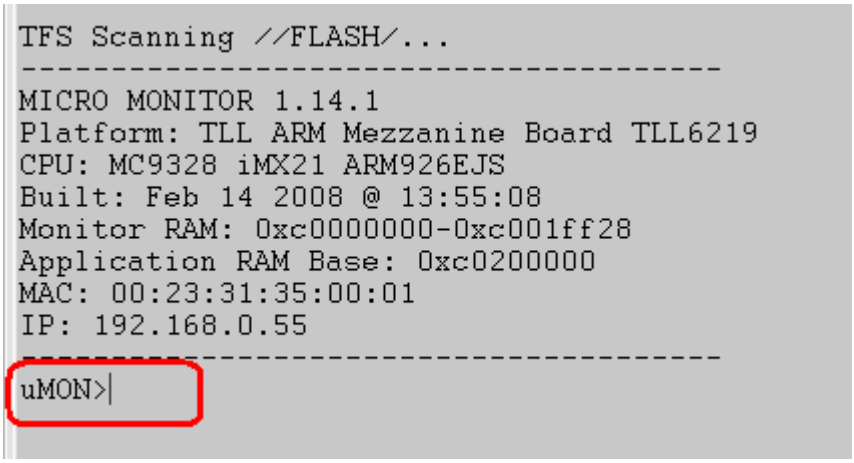
Note: Change COM1/ devttyS0 to appropriate COM Port for your PC. Check the COM port where the serial cable is connected. In general the available COM ports shall be checked with the following procedure in Windows Operating System

(Right click on My Computer → Properties → Hardware → Device Manager).

For Laptop Users USB-Serial should be detected. Screen shot shown below.

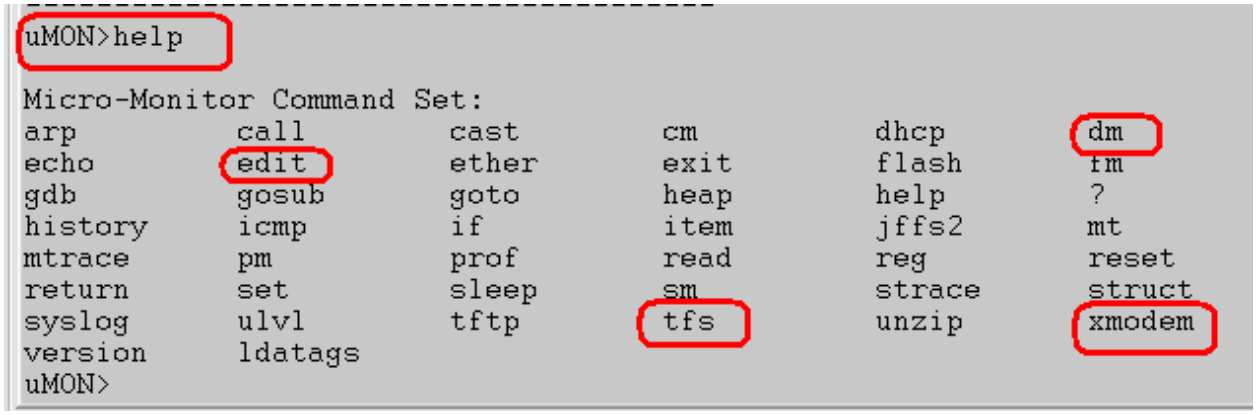


3. Press Enter on the terminal window to get uMON> prompt.



4. Type *help* or *?* on uMON prompt for help related to Micro-monitor commands

   uMON> *help*



It will display all the command set which uMON provides

For Example:

- tfs refers to tiny file systems and provides commands like tfs ls (for listing) tfs cp (for copying) etc
- dm refers to display memory and provides commands to view data at various memory locations.
- Xmodem is used for file transfer.
- Ether deals with commands related to Ethernet

Note: Refer to Micromonitor manual for further details on each command.

5. For every command, individual help commands can be typed. For example

   uMON> help xmodem

   uMON> help dm

```
uMON>help dm
Display Memory
Usage: dm -[24bdefl:msv:] {addr} [byte-cnt]
Options:
 -2   short access
 -4   long access
 -b   binary
 -d   decimal
 -e   endian swap
 -f   fifo mode
 -l#  size of line (in bytes)
 -m   use 'more'
 -s   string
 -v {var} load 'var' with element

Required user level: 0
uMON>
```

6. Type tfs ls for list on files in flash memory of TLL SILC-6219.
7. uMON> tfs ls

```
uMON>tfs ls
 Name                    Size   Location   Flags  Info
 LEDblink               36864  0xc829863c  E
 monrc                    133  0xc82a169c  e      envsetup
 startlinux              3072  0xc828e97c  e
 zImage               1169408  0xc81704bc

Total: 4 items listed (1209477 bytes).
uMON>
```

8. TLL SILC-6219 will come preloaded with four files as shown below
   - LEDblink → sample program to blink the user LEDs.
   - StartLinux → script to boot Linux
   - monrc → script to set parameters like IP address,Netmask etc.
   - zImage → the Linux image
9. Type help tfs to understand what each of flags mean.
10. uMON> help tfs
11. Type LEDblink on the prompt and notice the User LEDs are blinking.
12. uMON>LEDblink
13. Press Flash/Power Reset to return back to uMON> prompt.
14. Type startLinux (script) to boot the Linux.

15. uMON>startLinux

```
uMON>startlinux
* Loading Kernel: zImage
* Src: 0xc81704bc Dst: 0xc0200000
* Len: 1169408
*
* Cmdline Arguments:
ATAGS (1140 bytes) at 0xc0200000...
 Core (flags/pgsize/rootdev) = 0x0/0x1000/0x0
 Mem32 (size/offset) = 0x08000000/0xc0000000
 Serial (hi/lo) = 0x00000000/0x00000000
 Ramdisk (flags/size/start) = 0x0/0x0/0x0
 Initrd (size/start) = 0x0/0x0
 Cmdline = <ip=192.168.0.55:192.168.0.1:192.168.0.1:255.255.255.0::eth0:off mem=64M
console=ttyS0:230400 console=tty0  root=/dev/mtdblock1 rootfstype=jffs2 >
* Calling 0xc0201000 with args -1071644672
****************************************


Copying ATAGS page.
Uncompressing Linux.....................................|
```

16. This script will pass some parameters for Linux to boot. It will uncompress the Linux and load the kernel. After loading **/ #** can be seen on the terminal which is a standard Linux prompt or shell.
17. Once the Linux is booted type ls to get a directory listing.
18. / # *ls*

```
/ # ls
bin   dev   etc   lib   mnt   opt   proc  sbin  sys   tmp   usr   var
/ # help

Built-in commands:
------------------
     . : alias bg break cd chdir continue eval exec exit export false
     fg hash help jobs kill let local pwd read readonly return set
     shift times trap true type ulimit umask unalias unset wait [
     ash awk basename bunzip2 busybox bzcat cat chgrp chmod chown
     chroot clear cmp cp cut date dd df dirname dmesg dos2unix du
     echo env expr false fdisk fgrep find free freeramdisk grep gunzip
     gzip halt head hexdump hostid hostname hwclock id ifconfig ifdown
     ifup inetd init insmod ip kill killall klogd ln logger login
     losetup ls lsmod makedevs md5sum mesg mkdir mkfifo mknod mkswap
     modprobe more mount mv nslookup patch ping pivot_root poweroff
     printf ps pwd rdate reboot reset rm rmdir rmmod route rpm rx
     sed sh sleep sort strings stty swapoff swapon sync sysctl syslogd
     tail tar tee telnet telnetd test tftp time touch traceroute true
     tty umount uname uncompress uniq unix2dos unzip uptime usleep
     uudecode uuencode vconfig vi watch watchdog wget which who whoami
     yes zcat
```

19. /# *help* will display all the commands available with Linux.
20. Type ping command to verify Ethernet connectivity (Desktop IP)
21. /# ping 192.168.0.2

Type reboot command in Linux shell to get back uMON> prompt.

22. / # reboot
23. The TLL SILC-6219 is ready for use.

Chapter3 - SILC-6219 Subsystems

The block diagram for the SILC-6219 is shown below in figure 3.1. There are 8 basic subsystems on the board. These include:

1. Processor: iMX21
2. Memory: SDRAM and FLASH
3. Communications: USB, Ethernet, RS232
4. Display: LCD
5. LEDs, Jumpers and Switches
6. Glue Logic: CPLD, Mezzanine IO buffers, JTAG
7. General Purpose I/O
8. Power Supplies

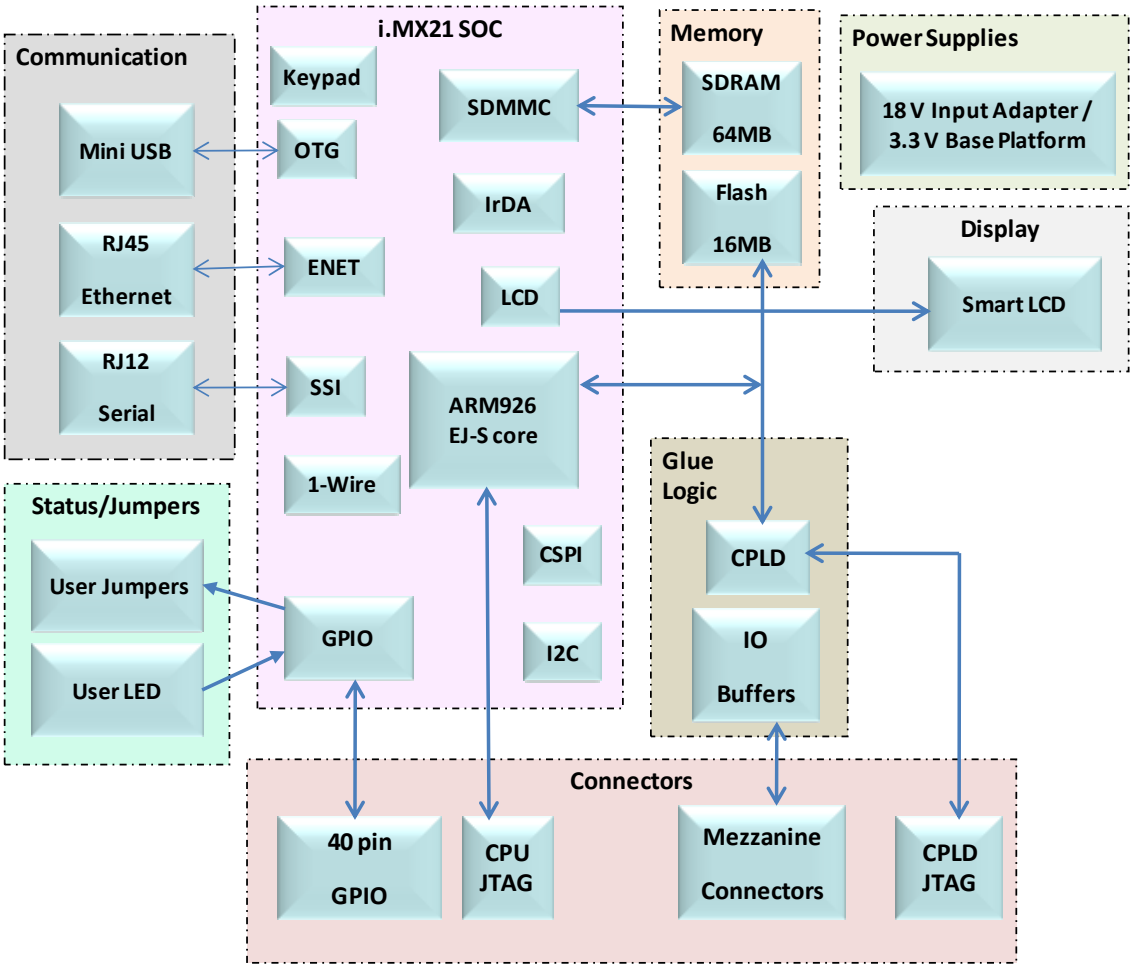


Figure 3.1

These subsystems are described below in sections 3.1 – 3.8

3.1 i.MX21 System-on-Chip

The i.MX21 SOC boasts a robust array of features that can support a wide variety of applications. Below is a brief description of the features.

- ARM926EJ-S™ Core Complex
- Enhanced Multimedia Accelerator (eMMA)
- Display and Video Modules
  - — LCD Controller (LCDC)
  - — Smart LCD Controller (SLCDC)
  - — CMOS Sensor Interface (CSI)
- Bus Master Interface (BMI)
- Wireless Connectivity
  - — Fast Infra-Red Interface (FIRI)
- Wired Connectivity
  - — USB On-The-Go (USBOTG) Controller
  - — Four Universal Asynchronous Receiver/Transmitters (UART*x*)
  - — Three Configurable Serial Peripheral Interfaces (CSPI*x*) for High Speed Data Transfer
  - — Inter-IC (I2C) Bus Module
  - — Two Synchronous Serial Interfaces (SSI) with Inter-IC Sound (I2S)
  - — Digital Audio Mux
  - — One-Wire Controller
  - — Keypad Interface
- Memory Expansion and I/O Card Support
  - — Two Multimedia Card and Secure Digital (MMC/SD) Host Controller Modules
- Memory Interface
  - — External Interface Module (EIM)
  - — SDRAM Controller (SDRAMC)
  - — NAND Flash Controller (NFC)
  - — PCMCIA/CF Interface
- Standard System Resources
  - — Clock Generation Module (CGM) and Power Control Module
  - — Three General-Purpose 32-Bit Counters/Timers
  - — Watchdog Timer
  - — Real-Time Clock/Sampling Timer (RTC)
  - — Pulse-Width Modulator (PWM) Module
  - — Direct Memory Access Controller (DMAC)
  - — General-Purpose I/O (GPIO) Ports
  - — Debug Capability

3.2 Memory Subsystem

The i.MX21 utilizes a 32-bit address bus that is capable of addressing a 4-Gbyte physical address space. This space is divided into sections of 512M byte regions within which various memories and peripherals are mapped. The memory map for the FLASH and DRAM memory is shown below in Figure 3.2. There are four primary blocks of external memory that can be accessed by the i.MX21. These accomplished via the 4 chip-select signals which are available to select the appropriate range of memory.

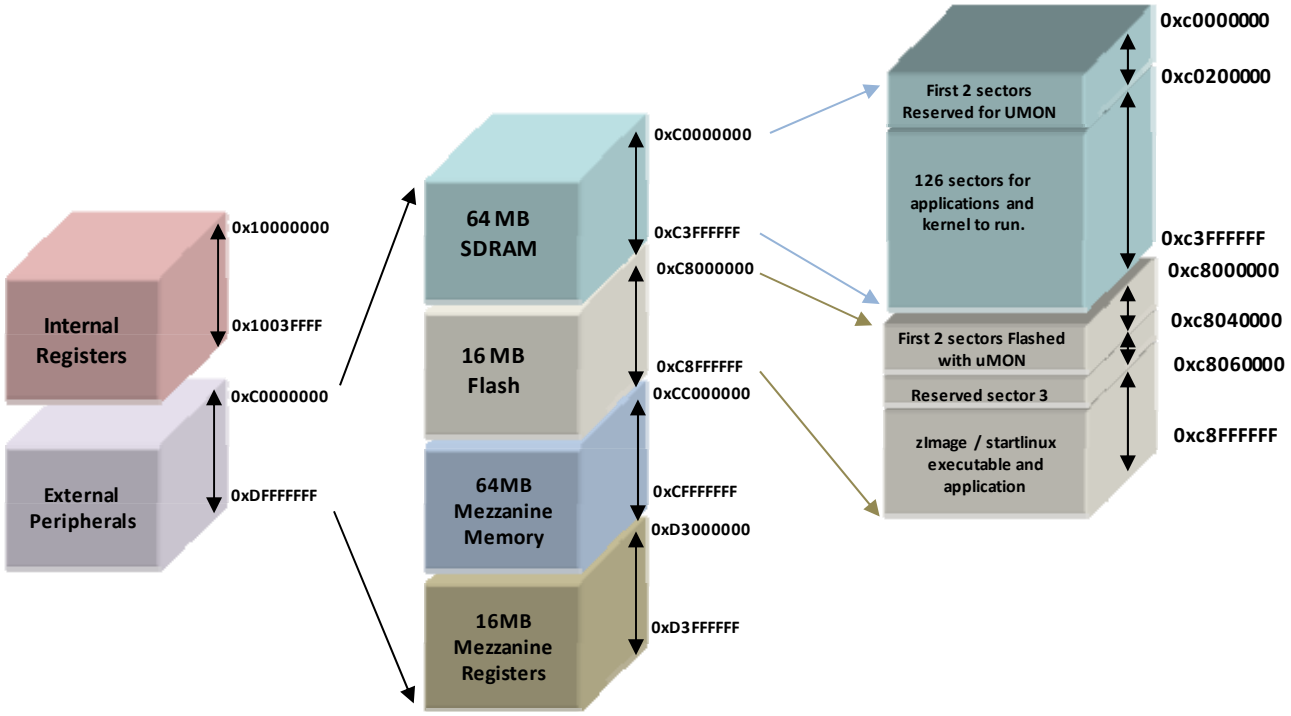


Figure 3.2 SILC-6219 Memory Map

CSD0 is used to select the 64 MB of DRAM. The range of addresses for the DRAM is from 0xC000_0000 – 0xc3FFF_FFFF. The DRAM is for execution of uMON and LINUX

CS0 is used to select the 16MB of NOR FLASH. The range of addresses for the FLASH is from 0xC800_0000 – 0xC8FFF_FFFF. The FLASH stores the binary images of uMON and Linux.

CS1 is used to select mezzanine memory. The CS1 signal is output to the SILC-5000 based board for use by FPGA to select 64MB of memory attached to the FPGA. CS1 decodes memory in the range of 0xCC00_0000 to 0xCFFF_FFFF.

CS5 is used to select mezzanine registers. The CS5 signal is output to the SILC-5000 based board for use by FPGA to select 16MB of registers in the FPGA. CS5 decodes memory in the range of 0xD300_0000 to 0xD3FF_FFFF.

3.3 Communications Subsystem

3.3.1 USB

The TLL SILC-6219 platform comes with a mini-USB OTG fully compatible with USB 2.0. USB can operate in both full speed (12 Mbps) and high speed (480 Mbps) utilizing a mini-USB standard 5 pin OTG connector.

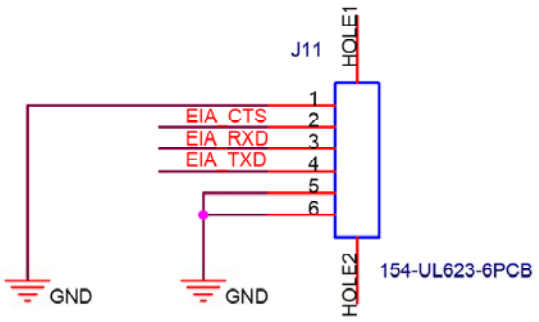
3.3.2 Ethernet

The TLL SILC-6219 platform provides Ethernet communication via a standard RJ45 connector. A LAN9115 Ethernet controller from SMSC is used. The controller supports a wide range of capabilities including:

- Full compliance with IEEE 802.3 and 802.3u standards
- 10BaseT and 100BaseT communications
- Full and half duplex support
- Full duplex flow control
- Preamble generation and removal
- Automatic 32-bit CRC generation
- Loop back modes

3.3.2 RS232

There is one RS-232 port which is accessible via an RJ-12 jack. The RJ-12 is a standard telephone jack 6pin connector. The cable required for the serial communication is a NULL modem cable (Tx → Rx, Rx→ Tx, Gnd → Gnd). The baud rate is initially set to 115K. uMON and Linux utilize this port for all communications. The interface is via TTL logic levels, therefore a TTL-RS232 level translator is required. Refer to Appendix-A for details on how to install the USB-SERIAL Cable Driver. The schematic figure to the right shows the pin configuration of the RJ12 jack



3.4 LCD Display

TLL6219 provides two LCD connectors. LCD can be connected upwards or can be flipped downwards. The TLL6219 LCD connectors can interface with SLCD (smart LCD) also. The SLCDC module has capability of directly interfacing with selected display devices, supporting both monochrome and color modes.

3.5 LEDS, Jumpers and Switches

3.5.1 User LEDs

Two user-LEDs (RED) are provided on the SILC-6219 platform. The PINS connected to LED are multiplexed with NAND flash controller and PCMCIA signals. The user LEDs can be used for indication purpose.

Processor BGA Pin	Signal Name	Description	Direction	GPIO Port F
M11	NFIO7 / PC_CD1	NAND Flash Controller /PCMCIA	B	PF14
V13	NFIO6 / PC_CD2	NAND Flash Controller /PCMCIA	B	PF13

3.5.2 User Jumpers

Two user jumpers are provided on the SILC-6219 platform. The switches are multiplexed with NAND flash controller and PCMCIA. They are in the form of jumpers placed next to the GPIO connector

Processor BGA Pin	Signal Name	Description	Direction	GPIO Port F
T11	NFIO5 / PC_WAIT (IDE_IORDY)	NAND Flash Controller /PCMCIA	B	PF12
U12	NFIO4 / PC_READY (IDE_INTRQ)	NAND Flash Controller /PCMCIA	B	PF11

NOTE: In the current implementation of uMON, the LEDs are illuminated if the jumpers are installed.

3.5.3 Boot Mode jumpers

There are three jumpers for setting the boot mode of i.MX21. The reference manual describes four jumpers for setting the boot mode. Out of the four the MSB Boot[3] is always connected to ground and is not brought out on SILC-6219 platform. These jumper settings provide various options to boot the ARM926 core based processor. Refer to GPCR (Global Peripheral Control Register) for details on boot options.

3.5.4 Reset Switch

There are two reset switches on the SILC-6219 board. The first one is connected to the POR circuitry as shown below in Figure 3.3. The other is connected to the RESET generator as shown below in Figure 3.4

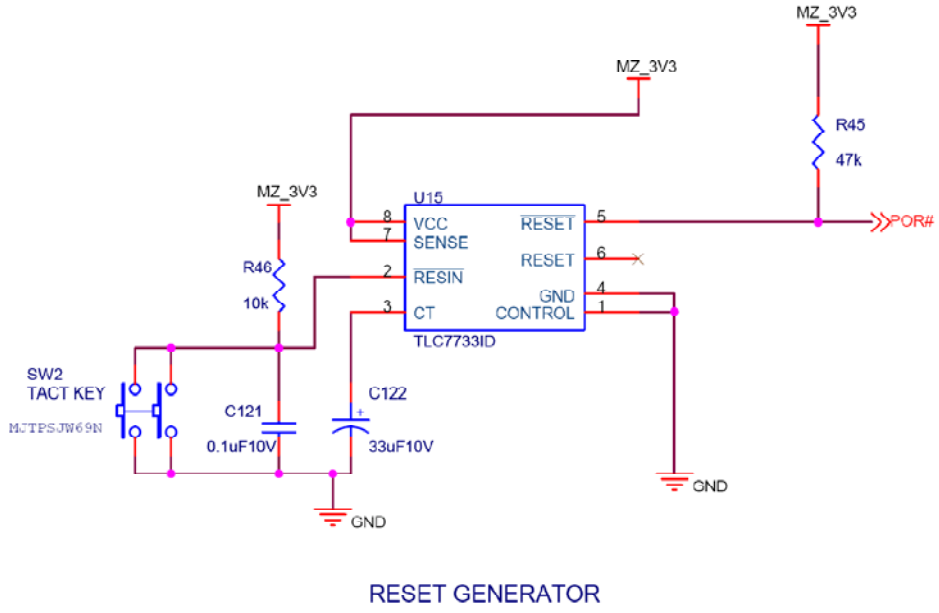


Figure 3.3 POR Generator

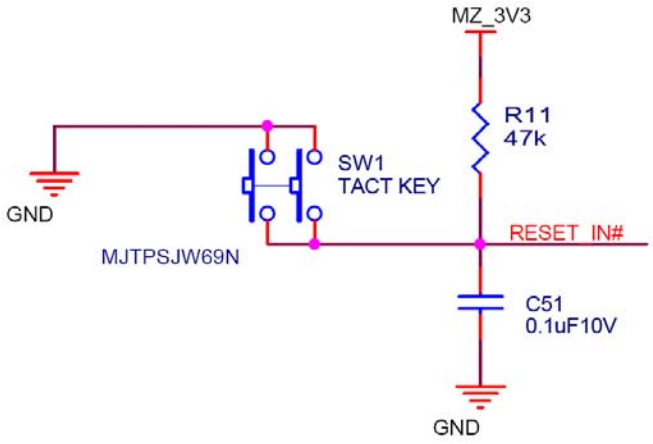


Figure 3.4 Reset Generator

3.6 Glue Logic

The glue logic on the SILC-6219 is used to control the SDRAM, FLASH and off board accesses via the mezzanine connectors. The glue logic is implemented with a programmable logic device from Xilinx

3.6.1 CPLD

A Xilinx 9572 CPLD is used to control the bus timing between the SILC-6219 and the SILC-5000 baseboard.

The READ CYCLE timing diagram for the mezzanine port is shown below in Figure 3.5:

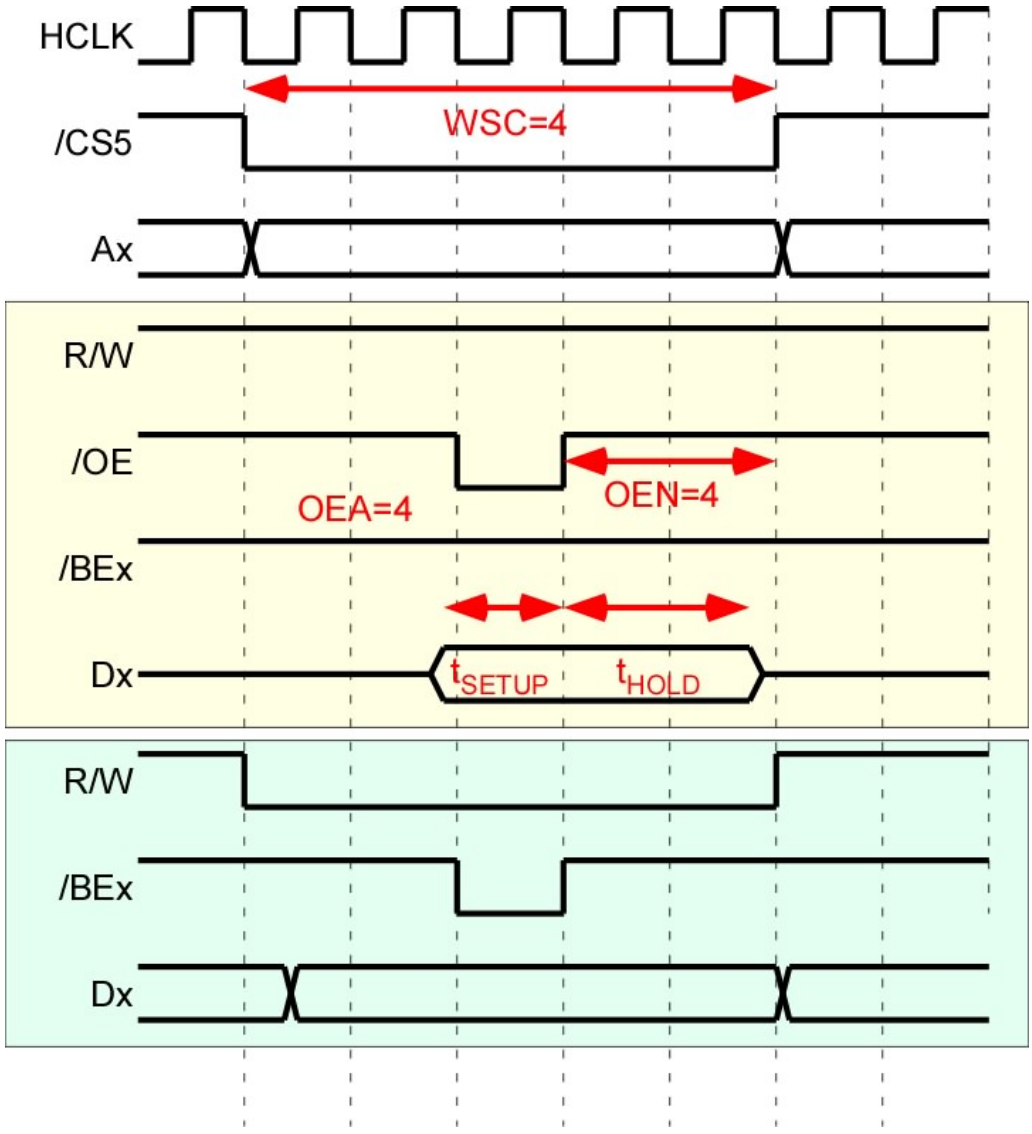


Figure 3.5 Mezzanine Bus READ Timing

The WRITE CYCLE bus timing diagram is shown below in Figure 3.6

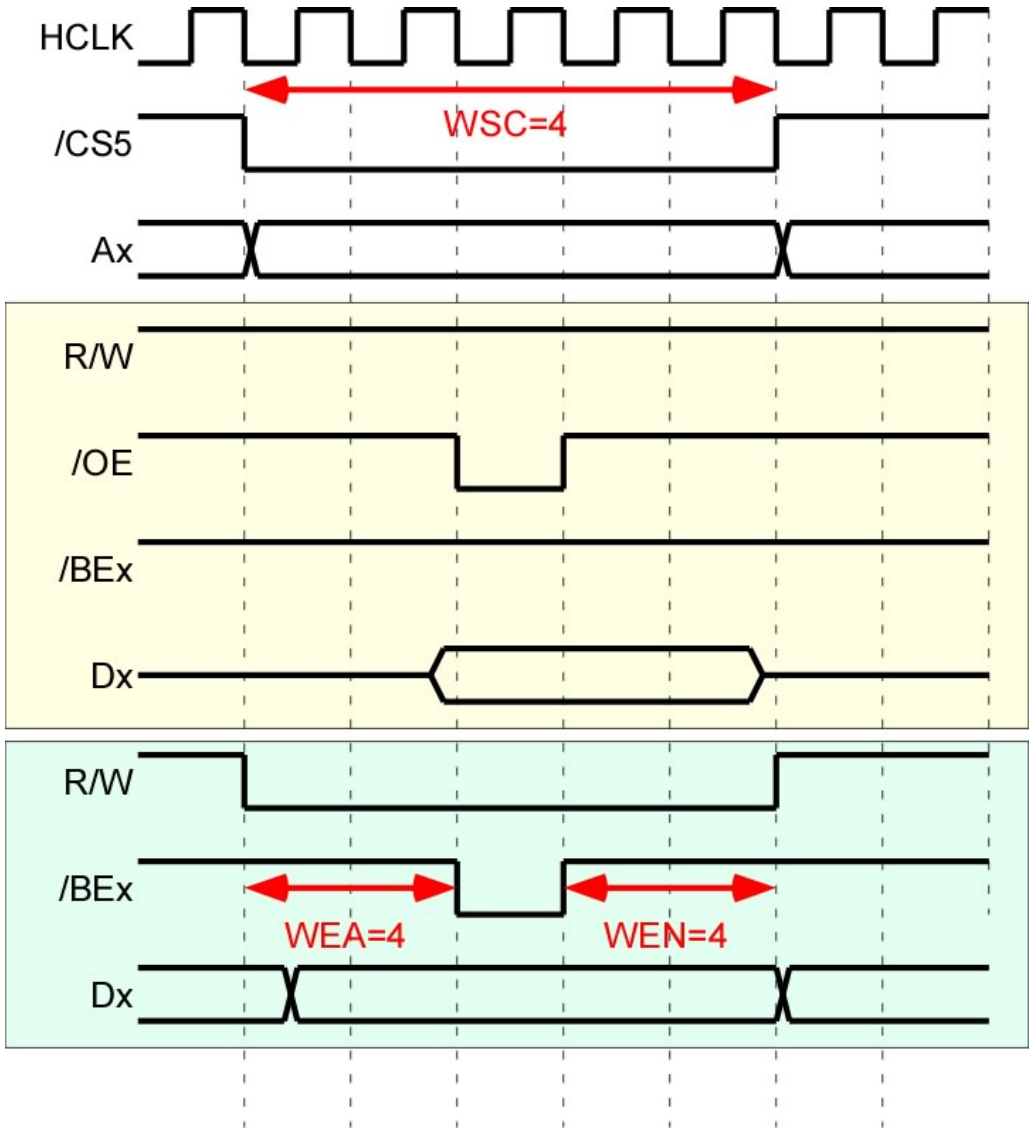


Figure 3.6 WRITE CYCLE Bus Timing Diagram

The SILC-6219 utilizes an asynchronous control mechanism to interface to the external peripherals connected to the mezzanine connectors. This is accomplished using the DTACK input signal. DTACK is used to externally terminate a data transfer. For DTACK enabled operations, a bus time-out monitor generates a bus error when an external bus cycle is not terminated by the DTACK input signal after 1024 HCLK clock cycles have elapsed, where HCLK is the internal system clock driven from the PLL module.

The pin configuration for the mezzanine connector is shown below in Table 3.1

Mezzanine Signal	Source	Description
MZ_BUF_D[31:0]	BUFFER	Buffered 32 bit Data Bus from i.MX21
MZ_BUF_A[23:0]	BUFFER	Buffered 24 bit Address Bus from i.MX21
MZ_CPLD_BYTE[3:0]	CPLD	Buffered Byte-Enable signals from i.MX21
MZ_CPLD_RW	CPLD	Read/Write signal
MZ_CPLD_AS	CPLD	Address Strobe Signal
MZ_CPLD_RESET	CPLD	Reset-Out signal from i.MX21
MZ_CPLD_MISC[14:0]	CPLD	Miscellaneous control signals to SILC-5000 Baseboard
MZ_CPLD_CLKO	CPLD	Buffer i.MX21 Bus Clock (HCLK)
FPGA_MZ_DTACK	FPGA	DTACK signal from SILC-5000 Baseboard

Table 3.1 Mezzanine Connector Signal Description

3.6.2 JTAG

A 20 pin CPU JTAG connector is used for flashing the TLL SILC-6219 system. It also provides standard debug access to ARM926 core and executes the instructions independently. The main role of this JTAG connection is to program the CPLD on the platform. The figure to the right illustrates the pin configuration used by the JTAG connector.

3.7 General Purpose Input-Output (GPIO)

The i.MX21 processor provides six GPIO ports which are multiplexed with one or more dedicated functions. Each port has 17 registers. These register settings decide if the pin is used for primary function (say- CSPI2_MOSI) or it used for secondary function (USBH2_TXDP) or it used by the GPIO port. The 40 pin GPIO connector includes 36 GPIO pins from the i.MX21for I2C, CSPI, UART, Camera sensor interface as indicated below:

Connector Pin	BGA Pin	Signal Name	Description	Direction	GPIO Port
1	A8	CSI_D0	CMOS Sensor	I	PB10
2	C8	CSI_D1	CMOS Sensor	I	PB11
3	D9	CSI_D2	CMOS Sensor	I	PB12
4	G9	CSI_D3	CMOS Sensor	I	PB13
5	B9	CSI_D4	CMOS Sensor	I	PB14
6	H9	CSI_D5	CMOS Sensor	I	PB17
7	B10	CSI_D6	CMOS Sensor	I	PB18
8	D10	CSI_D7	CMOS Sensor	I	PB19
9	C10	CSI_HSYNC	CMOS Sensor	I	PB21
10	A10	CSI_VSYNC	CMOS Sensor	I	PB20
11	A9	CSI_PIXCLK	CMOS Sensor	I	PB16
12	C9	CSI_MCLK	CMOS Sensor	O	PB15
13	G18	UART2_CTS# /KP_COL7	UART /Keypad	O	PE3
14	J16	UART2_RTS# /KP_ROW7	UART /Keypad	I	PE4
15	M16	UART2_RXD/KP_ROW6	UART /Keypad	I / B	PE7
16	L16	UART2_TXD/KP_COL6	UART /Keypad	O/ B	PE6
17	B18	I2C_DATA	Inter IC communication	B	PD17
18	C18	I2C_CLK	Inter IC communication	B	PD18
19	H19	PC_SPKOUT/PWMO/TOUT2/TOUT3	Pulse Width Modulation	O	PE5
20	R16	SD1_CLK/CSPI3_SCLK	Secure Digital Host controller / Configurable serial Peripheral interface	B / O	PE23
21	P17	SD1_CMD/CSPI3_MOSI		B / O	PE22
22	N16	SD1_DO/CSPI3_MOSI		B / I	PE18

23	N18	SD1_D1	Secure Digital	B	PE19
24	P16	SD1_D2	Secure Digital	B	PE20
25	T17	SD1_D3/CSPI3_SS	Secure Digital / CSPI	B / O	PE21
26	D19	CSPI2_SCLK/USBH2_OE#	Configurable Serial Peripheral Interface / USB	B	PD22
27	E19	CSPI2_MOSI/USBH2_TXDP		B	PD24
28	E17	CSPI2_MISO/USBH2_TXDM		B	PD23
29	C19	CSPI2_SS1/USBH2_RXDP		B	PD20
30	D18	CSPI2_SS0/USBH2_FS		B	PD21
31	A16	SSI1_CLK	Synchronous Serial interface	B	PC23
32	B19	CSPI2_SS2/USBH2_RXDM	CSPI	B	PD19
33	C15	x_SSI1_TXDAT	Synchronous Serial interface	B	PC22
34	D16	x_SSI1_RXDAT		B	PC21
35	B15	x_SSI1_FS		B	PC20
36	A14	x_TOUT/SYS_CLK1/SYS_CLK2		O	PC14
37	-	GND	-	-	-
38	-	GND	-	-	-
39	-	GND	-	-	-
40	-	GND	-	-	-

3.8 Power supply

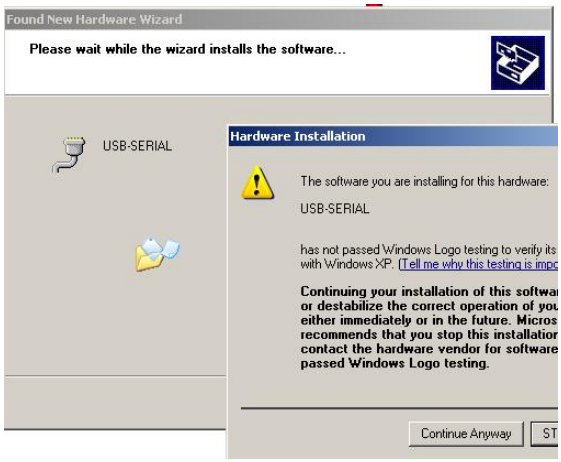
The SILC-6219 platform requires **+18V, 1A** power supply from adapter provided along with the package. The power supply adapter is 18V/1A and can work with 110 and 220 supply voltage. The power supply connector is **J15** with G and + written on GND and POSITIVE supply sides respectively. The green LED indicates proper power supply.

Note: The platform should be powered first before connecting any cables and should be removed last when removing all cables

APPENDIX – A

A.1 USB-Serial Cable Driver (for Windows OS):

- Connect only the USB-serial cable to your PC. Don't connect it with the board.
- Found New Hardware wizard will start.
- Put the CD-Rom for USB-serial driver installation.
- Use the recommended option for driver installation.



- Check in the device manager. It would have detected USB-SERIAL (COMx).x denotes any number of the COM port.

Note: There is no need for any Installation in Linux Operating System with Kernel 2.6.X. Connect the USB to serial cable hardware to the USB port of the PC and then use the Minicom application with **ttyUSB0** port

Appendix B – ARM 926EJS

The ARM926EJ-S™ is a member of the ARM9 family of general-purpose microprocessors. The processor is targeted at multi-tasking applications where full memory management, high performance, low die size, and low-power are essential for mobile computing. The ARM926EJ-S processor supports 16-bit and 32-bit ARM Thumb® instruction sets. These instruction sets allows the processor to be utilized in applications that need to balance high performance execution and high code density. The features include:

- ARM926EJ-S™ microprocessor core
  - — 16K instruction cache and 16K data cache
  - — High-performance ARM® 32-bit RISC engine
  - — Thumb® 16-bit compressed instruction set for a leading level of code density
  - — Efficient execution of Java byte codes
  - — EmbeddedICE™ JTAG software debug
  - — 100 percent user code binary compatibility with ARM7TDMI®
  - — Advanced Microcontroller Bus Architecture (AMBA™) system-on-chip multi-master bus interface
  - — Support for mixed loads of real-time and user applications via cache locking
  - — Virtual Memory Management Unit (VMMU)
- ARM Interrupt Controller (AITC)
  - — The AITC is connected to the primary AHB as a slave device and provides support for up to 64 interrupt sources. It generates normal and fast interrupts to the processor core.
- Digital Phase-Locked Loops (DPLLs) and Power Control Module
  - — Digital phase-locked loops (DPLLs) and clock controller for all internal clock generation
  - — MCUPLL generates system and CPU clocks from a 26MHz crystal
  - — USBPLL generates 48 MHz clock for the USB OTG from either a 26 MHz crystal or 32kHz
  - — Support for three power modes for different power consumption needs: run, doze, and stop.
- AHB to IP bus interfaces (AIPIs)
  - — Provide a communication interface between the high-speed AHB to a lower-speed IP bus for slave peripherals
- The Multi-Layer 6 × 4 AHB Crossbar Switch
  - — The crossbar switch allows for concurrent transactions to proceed from any input port (bus master) to any output port (bus slave
- CPU and System speed
  - — ARM926EJ-S core: up to 266 MHz
  - — System Clock: up to 133 MHz

Figure 1.0 below shows the block diagram of the i.MX21 SOC:

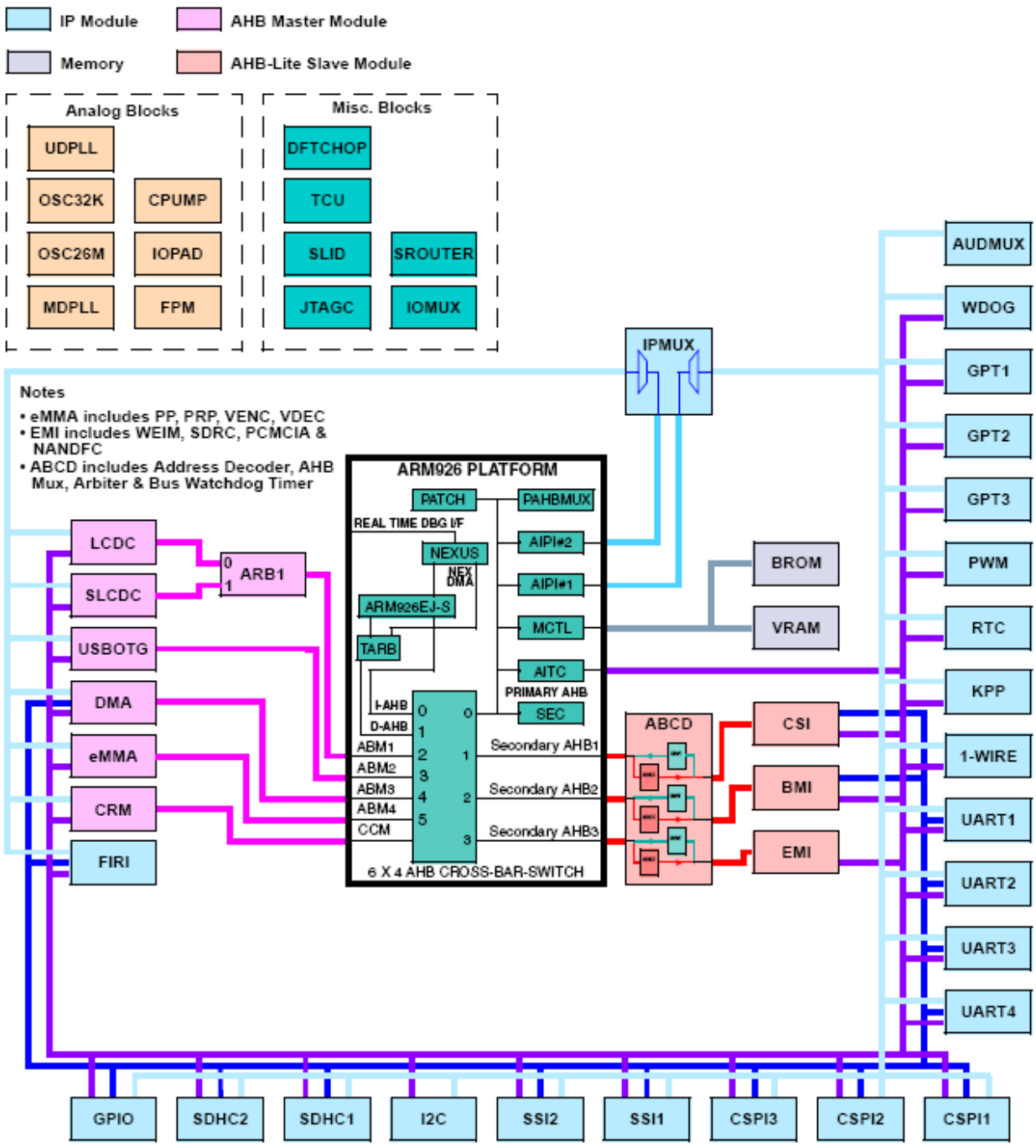


Figure 1.0 i.MX21 Block Diagram

B.1.1 ARM General Purpose Registers

The ARM926EJ-S™ has 31 general-purpose 32-bit registers. At any one time, 16 of these registers are visible. The other registers are used to speed up exception processing. All the register specifiers in ARM instructions can address any of the 16 visible registers.

The main bank of 16 registers is used by all unprivileged code. These are the User mode registers. User mode is different from all other modes as it is unprivileged, which means:

- User mode can only switch to another processor mode by generating an exception. The SWI instruction provides this facility from program control.
- Memory systems and coprocessors might allow User mode less access to memory and coprocessor functionality than a privileged mode.

Three of the 16 visible registers have special roles:

- **Stack pointer** Software normally uses R13 as a *Stack Pointer* (SP). R13 is used by the PUSH and POP instructions in T variants, and by the SRS and RFE instructions from ARMv6.
- **Link register --** Register 14 is the *Link Register* (LR). This register holds the address of the next instruction after a Branch and Link (BL or BLX) instruction, which is the instruction used to make a subroutine call. It is also used for return address information on entry to exception modes. At all other times, R14 can be used as a general-purpose register.
- **Program counter** Register 15 is the *Program Counter* (PC). It can be used in most instructions as a pointer to the instruction which is two instructions after the instruction being executed. In ARM state, all ARM instructions are four bytes long (one 32-bit word) and are always aligned on a word boundary. This means that the bottom two bits of the PC are always zero, and therefore the PC contains only 30 non-constant bits. Two other processor states are supported by some versions of the architecture. Thumb® state is supported on T variants, and Jazelle® state on J variants. The PC can be halfword (16-bit) and byte aligned respectively in these states.

The remaining 13 registers have no special hardware purpose. Their uses are defined purely by software.

B.1.2 Endianess

The SILC-6219 uses a little endian memory system. In a *little-endian* memory system:

- a byte or halfword at a word-aligned address is the least significant byte or halfword within the word at that address
- a byte at a halfword-aligned address is the least significant byte within the halfword at that address.

B.1.3 ARM Instruction Set Overview

The ARM instruction set can be divided into six broad classes of instruction:

- Branch instructions
- Data-processing instructions
- Status register transfer instructions
- Load and store instructions
- Coprocessor instructions
- Exception-generating instructions

Most data-processing instructions and one type of coprocessor instruction can update the four condition code flags in the CPSR (Negative, Zero, Carry and oVerflow) according to their result.

Almost all ARM instructions contain a 4-bit *condition* field. One value of this field specifies that the instruction is executed unconditionally. Fourteen other values specify *conditional execution* of the instruction. If the condition code flags indicate that the corresponding condition is true when the instruction starts executing, it executes normally. Otherwise, the instruction does nothing. The 14 available conditions allow:

- tests for equality and non-equality
- tests for <, <=, >, and >= inequalities, in both signed and unsigned arithmetic
- each condition code flag to be tested individually.

The instruction format is shown in table B.1 below.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Instruction Type
Condition				0	0	I	OPCODE				S	Rn				Rs				OPERAND-2												Data processing
Condition				0	0	0	0	0	0	A	S	Rd				Rn				Rs				1	0	0	1	Rm				Multiply
Condition				0	0	0	0	1	U	A	S	Rd HIGH				Rd LOW				Rs				1	0	0	1	Rm				Long Multiply
Condition				0	0	0	1	0	B	0	0	Rn				Rd				0	0	0	0	1	0	0	1	Rm				Swap
Condition				0	1	I	P	U	B	W	L	Rn				Rd				OFFSET												Load/Store - Byte/Word
Condition				1	0	0	P	U	B	W	L	Rn				REGISTER LIST																Load/Store Multiple
Condition				0	0	0	P	U	1	W	L	Rn				Rd				OFFSET 1				1	S	H	1	OFFSET 2				Halfword Transfer Imm Off
Condition				0	0	0	P	U	0	W	L	Rn				Rd				0	0	0	0	1	S	H	1	Rm				Halfword Transfer Reg Off
Condition				1	0	1	L	BRANCH OFFSET																								Branch
Condition				0	0	0	1	0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	1	Rn				Branch Exchange
Condition				1	1	0	P	U	N	W	L	Rn				CRd				CPNum				OFFSET								COPROCESSOR DATA XFER
Condition				1	1	1	0	Op-1				CRn				CRd				CPNum				OP-2			0	CRm				COPROCESSOR DATA OP
Condition								OP-1			L	CRn				Rd				CPNum				OP-2			1	CRm				COPROCESSOR REG XFER
Condition				1	1	1	1	SWI NUMBER																								Software Interrupt

Table B.1 ARM Instruction Set Format

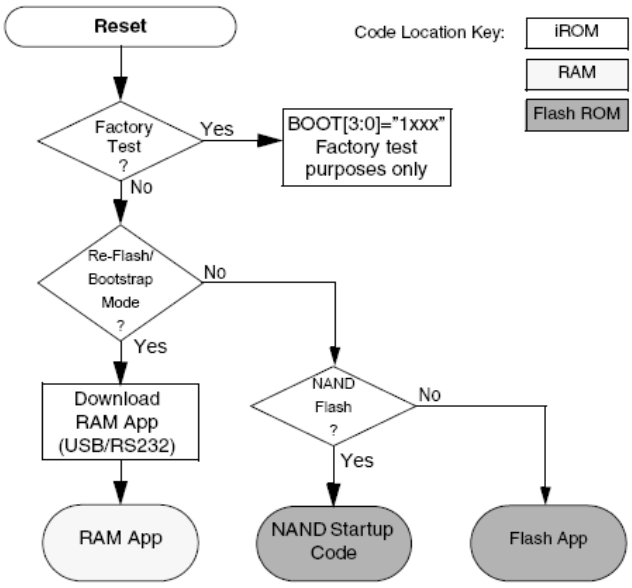
B.1.4 Boot Manager

The system boot-up is designed according to the configuration of the external BOOT pins, and hence sub-divided into Boot-external, Boot-internal, In-Factory test, Bootstrap mode and Normal flash boot-up modes. The i.MX21 processor comes with a 24 Kbyte internal ROM (iROM) which contains a System Boot Manager to process the serial bootstrap operations according to all possible scenarios from power-up. The iROM is controlled by various pins and laser fuse configurations. Table B.2 shows the boot-up modes for the various configurations of the BOOT pins.

Inputs BOOT[3:0]	Output Signals Active Device	Boot Address
0000	iROM (Bootstrap USB/UART)	0x00000030
0001	iROM (Bootstrap USB/UART)	0x00000030
0010	iROM (8-bit 2 Kbyte NAND Flash)	0xDF003000
0011	iROM (16-bit 2 Kbyte NAND Flash)	0xDF003000
0100	iROM (16-bit 512 byte NAND Flash)	0xDF003000
0101	iROM (16-bit CS0 at D[15:0] (NOR Flash))	0xC8000000
0110	iROM (32-bit CS0 at D[31:0] (NOR Flash))	0xC8000000
0111	iROM (8-bit 512 byte NAND Flash)	0xDF003000

Table B.2 BOOT pin configuration

The following diagram shows the boot-up flow diagram for the i.MX21:



B.1.5 JTAG Controller

The JTAG Controller module supports debug access to ARM926 core. The TAP ports—TCK, TDI, TMS, TRST and TDO—are not multiplexed with scan chains and are not used for any scan control. The JTAG Controller is compatible with IEEE1149.1 Standard Test Access Port and Boundary Scan Architecture, but without the logic and instructions needed for Boundary scan.

The debug features of JTAG provide the following capabilities:

- Provide debug access to the ARM926 core and execute its specific JTAG instructions independently.
- Provide ability to program the Flash memory on the SILC-6219 via Macraigor tool set.

B.2 Clock Generation

There are two clock controller modules in the i.MX21: the ARM9 platform clock controller and the PLL Clock Controller module which produces the clock signals used and distributed by the ARM9 platform clock controller. The PLL Clock Controller generates clock signals used throughout the i.MX21 chip and also for external peripherals. The PLL Clock Controller also serves as the interface between the ARM9 platform and the peripherals on the i.MX21. The primary function of the ARM9 platform clock controller is to take the clock signals from the PLL Clock Controller distribute them to various peripherals on the ARM9 platform. The clock control module contains the logic to turn clocks on or off and determine when the ARM9's clock can be turned off. This module also synchronizes the JTAG interface to the CLK domain. The ARM9 platform clock controller is not a user programmable or accessible module where the PLL Clock Controller is. Further details on how to program the clock controller is located in the MC9328MX21 Programmers Manual.

B.2.1 Clock Distribution

There are two DPLLs in the PLL Clock Controller: the MCU/System PLL (MPLL) and the Serial Peripheral PLL (SPLL) use digital and mixed analog/digital circuits to provide clock frequencies for wireless communication and other applications. The MPLL primarily generates the CLK signal to the ARM9 and HCLK (also called System clock) for the system bus and for most of the on-chip peripherals including the clock for LCDC pixel clock, NAND Flash Controller clock and FIRI MIR clock. The SPLL produces the primary clock to the clock dividers for USB OTG, SSI1, SSI2 and the FIRI FIR clock.

Both DPLLs (MPLL and SPLL) accept either the output of the FPM or the OSC26M as a source from which to generate the required frequencies for ARM9 platform and/or peripherals using a fractional frequency multiplication method.

To produce the wide range of on-chip clock frequencies required by the i.MX21, the core clock generator uses a two-stage phase locked loop. The first stage is a Frequency Pre-Multiplier PLL (FPM) which multiplies the input frequency by a factor of 512. If the input crystal frequency is 32.768 kHz, the pre-multiplier multiplies it by a factor of 512 to 16.78 MHz (16.384 MHz. for a 32 kHz crystal). The output of the FPM is one of the clock sources for the MPLL and SPLL. Power management of i.MX21 is accomplished by controlling the clock output of the MPLL and SPLL units. The distribution of clocks in the i.MX21 is shown in the general block diagram of the entire module figure 4.7 below.

There are two external clock sources to the PLL Clock Controller:

- 32 kHz external crystal
- 26 MHz external source/crystal

Settings in the Clock Source Control Register (CSCR) are used to independently configure the external clock sources applied to the MPLL and SPLL.

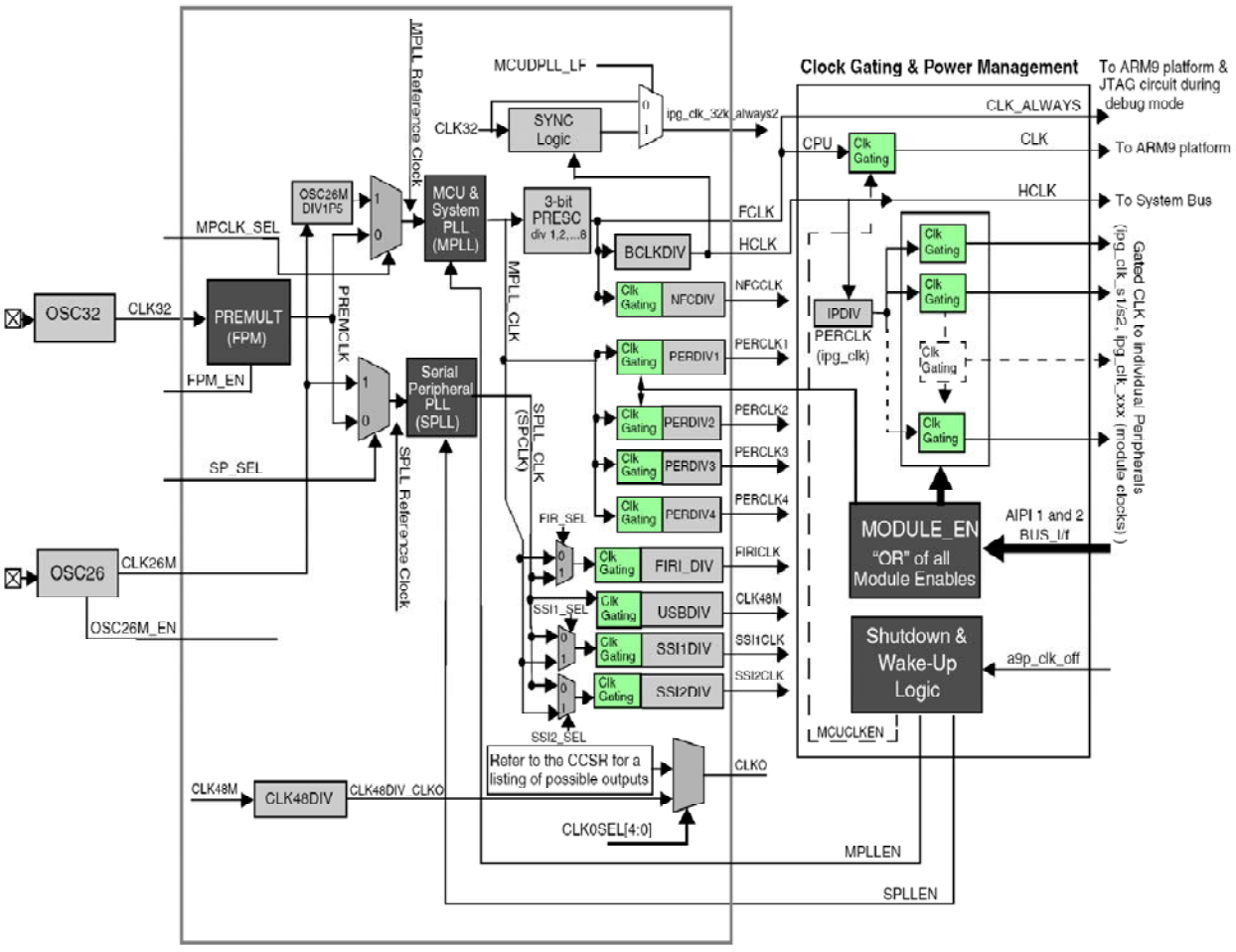


Figure B.1 i.MX21 Clocking Block Diagram

The i.MX21 can use either a 32 kHz or a 32.768 kHz crystal as the external low frequency source. The SILC-6219 uses a 32.768 KHz crystal. The signal from the external 32.768 KHz crystal is the source of the CLK32 signal that is sent to the real time clock (RTC). The output of the 32.768 KHz crystal is also input to the FPM (Frequency Pre-Multiplier) to produce the 16.384 MHz signal that is input to the DPLLs. The output of the MPLL is sent to the prescaler (PRESC) module to produce the clock (CLK) signal for the ARM core as well as the HCLK for the system bus and module operation.

B.2.2 Clock Configuration

The clock configuration on the SILC-6219 is programmed in the uMon monitor program during power-on-reset (POR) using the following ARM assembly code:

```
init_pll:

// ------------------------------------------------------------------
// Setup the MPLL
// ------------------------------------------------------------------
        ldr r2, =0x01381cc5     // 256 MHz
        ldr r1, =0x10027004
        str r2, [r1]
        bl     delay_200
// ------------------------------------------------------------------
// Set SPLL to default - 288 MHz
// ------------------------------------------------------------------
        ldr r2, =0x0272216d
        ldr r1, =0x1002700c
        str r2, [r1]
// ------------------------------------------------------------------
//   Set HCLK to 64 MHz
// ------------------------------------------------------------------
        ldr r2, =0x17000e07     // 256 MHZ CPU clock divide by 4
        ldr r1, =0x10027000
        str r2, [r1]
// ------------------------------------------------------------------
//   Set PERCLK1 to 16 MHz
// ------------------------------------------------------------------

        ldr r2, =0x0307070f    // 256 MHz CPU Clock divide by 16
        ldr r1, =0x1002701c           // Write to PCDR1
        str r2, [r1]
        bl     delay_200
// ------------------------------------------------------------------
// Restart the PLLs
// ------------------------------------------------------------------
        ldr r2, =0x17600e07    // 240/256/266 MHz CPU Clock
        ldr r1, =0x10027000
        str r2, [r1]
```


B.3 Interrupt Controller

The ARM926EJ-S™ Interrupt Controller (AITC) is a 32-bit peripheral which collects interrupt requests from up to 64 sources and provides an interface to the ARM926EJ-S core. The AITC includes software controlled priority levels for normal interrupts. The AITC block diagram is shown in figure 4.8 below.

The AITC performs the following functions:

- Supports up to 64 interrupt sources
- Supports fast and normal interrupts
- Selects normal or fast interrupt request for any interrupt source
- Indicates pending interrupt sources via a register for normal and fast interrupts
- Indicates highest priority interrupt number via register (can be used as a table index)
- Independently enable or disable any interrupt source
- Provides a mechanism for software to schedule an interrupt
- Supports up to 16 software controlled priority levels for normal interrupts and priority masking

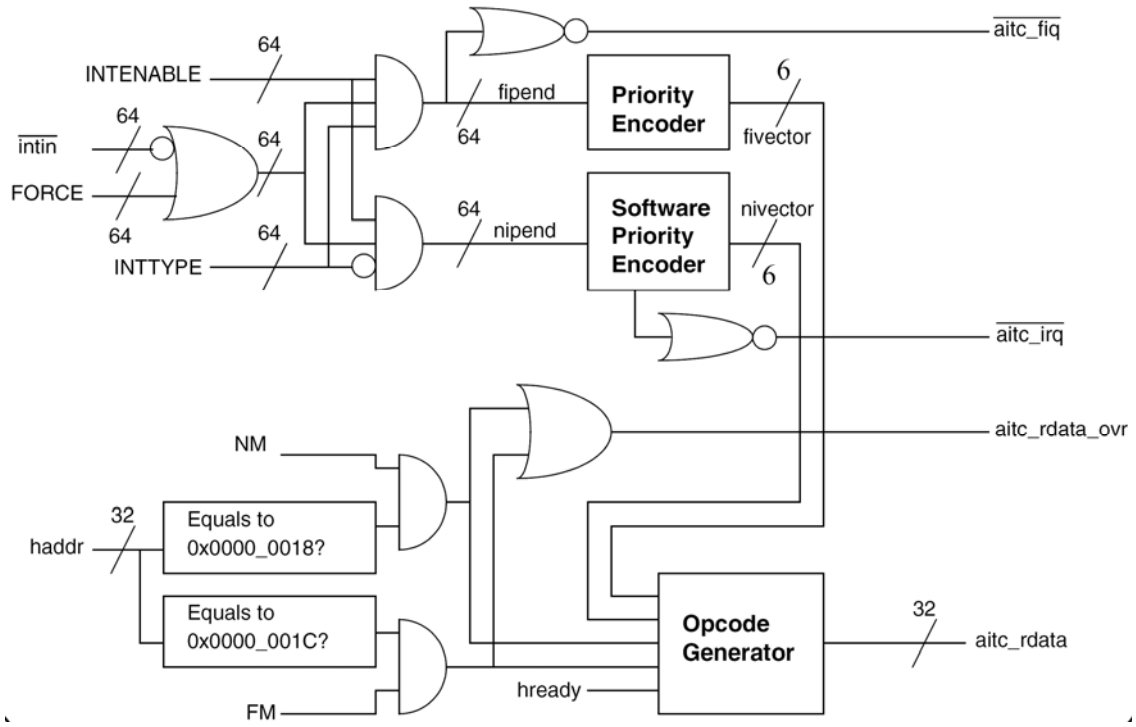


Figure B.2 i.MX21 ARM Interrupt Controller

The interrupt controller consists of a set of control registers and associated logic to perform interrupt masking, and priority support of normal interrupts.The interrupt source registers INTSRCH / INTSRCL) are a pair of 32-bit status registers with a single interrupt source associated with each of the 64 bits. An interrupt line or set of interrupt lines are routed from each interrupt source to the INTSRCH or INTSRCL register. This allows up to 64 distinct interrupt sources in an implementation. Interrupt requests may be forced to be asserted by way of the interrupt force registers (INTFRCH / INTFRCL). Each bit in this register is logically “OR-ed” with the corresponding hardware request line prior to feeding the INTSRCH or INTSRCL register inputs.

There is a corresponding set of interrupt enable registers (INTENABLEH / INTENABLEL), also 32-bits wide which allow individual bit masking of the INTSRCH / INTSRCL registers. There is also a corresponding set of interrupt type register (INTTYPEH / INTTYPEL) which selects whether an interrupt source will generate a normal or fast interrupt to the ARM926EJ-S core.

The normal interrupt vector register (NIVECSR) indicates the vector index of highest priority pending normal interrupt. The 64 interrupt sources are assigned from 0-63 respectively. INT-8 (GPIO interrupt) is assigned interrupt numbers 64-255. To determine which interrupt sources Linux recognizes type the following command in Linux

more /proc/interrupts

On the TLL-6219 you will see the following 4 interrupt sources:

```
Int #       #INT's      Source
 20:        117         IMX-uart
 26:        15985       i.MX Timer Tick
 55:         0          imx21-hc:usb1
224:         2          smsc911x
```

The first 3 sources are internal interrupts. INT-224 is an external interrupt from the Ethernet Controller. The ENET interrupt is connected to Port-F Pin-0 (PF0). This corresponds to bit #160 in the GPIO bit ordering. To calculate the interrupt number:

INT Number = 64 + 160 = 224

There is one external interrupt that is input to Port F pin 16 (PF16). This translates to interrupt number 240.

Appendix-C Subsystem Schematics

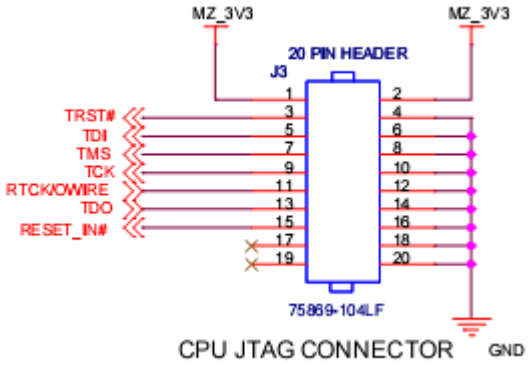
C.1 Power supply

CON-2 is used to provide 18V to the SILC-6219 board

LED3V3 will illuminate when the power regulator is generating 3.3V

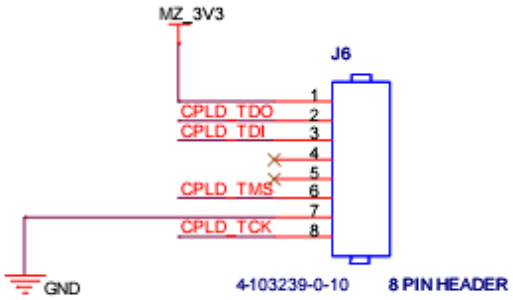
C.2 CPU JTAG

This is a 20 pin JTAG connector used for flashing the TLL SILC6219 system. Provides standard debug access to ARM926 core and executes the instructions independently.



C.3 CPLD JTAG

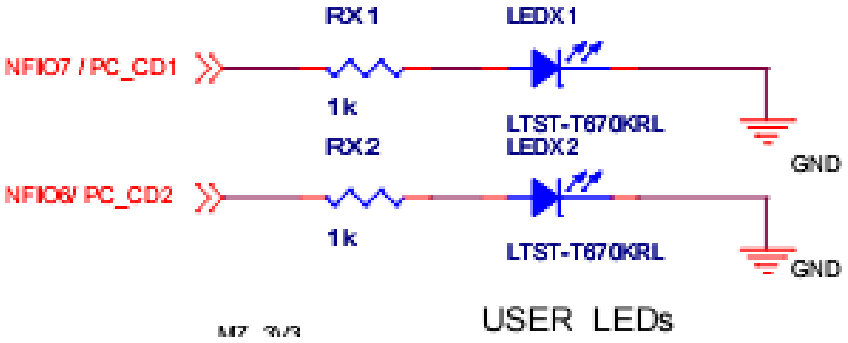
This is an 8 pin CPLD JTAG connector used to configure CPLD on TLL6219 platform. The Xilinx IMPACT tool is used to transfer the .jed (top.jed are we providing this file to the user, if yes provide the path the in the CD or remove the reference) file to the CPLD. Here CPLD is used basically to perform address decoding logic / glue logic.



CPLD JTAG CONNECTOR

C.4 User LEDs

Two user-LEDs (RED) are provided on the TLL6219 platform. The Pins connected to LED are multiplexed with NAND flash controller and PCMCIA card.



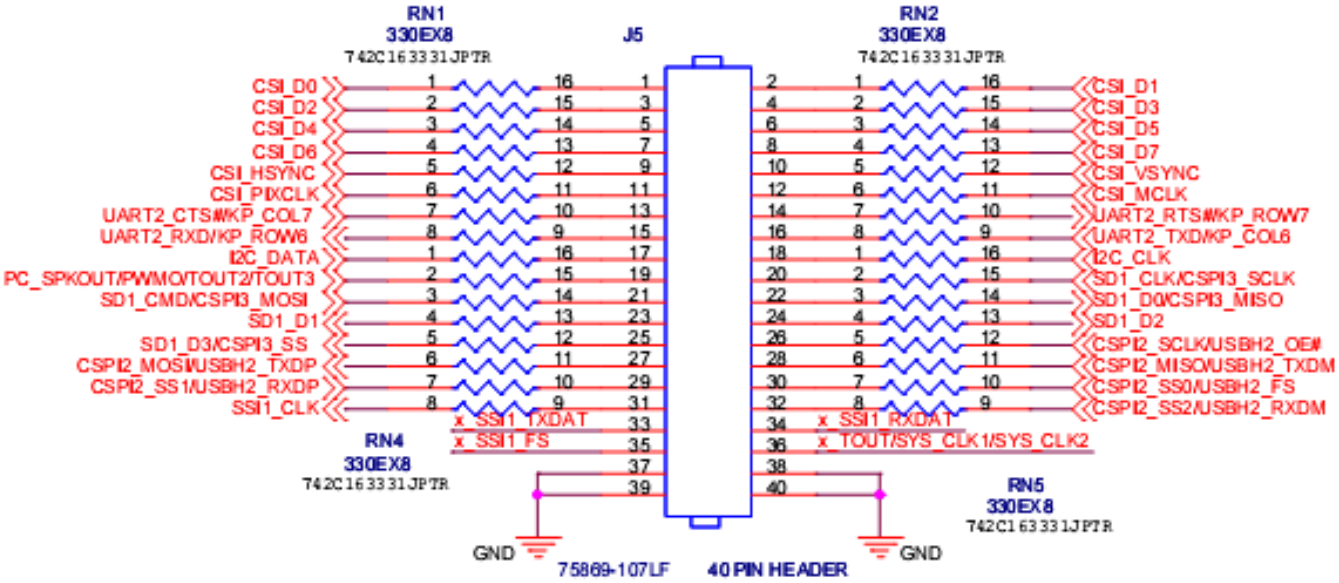
USER LEDs

C.5 User switches

Two user Switches are provided on the TLL6219 platform. The switches are multiplexed with NAN flash controller and PCMCIA. They are in the form of jumpers.

C.6 GPIO Connector

I.MX21 family provides six GPIO ports which may be multiplexed with one or more dedicated functions. The SILC-6219 brings out selected pins from the six GPIO ports on the i.MX21 to a 40 PIN connector.

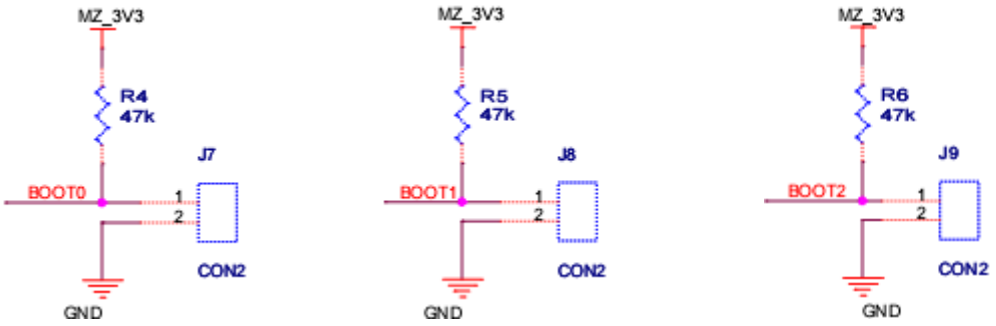


GPIO CONNECTOR

C.7 Boot Mode jumpers

There are three jumpers for setting the boot mode of i.MX21. The reference manual describes four jumpers for setting the boot mode. Out of the four the MSB Boot[3] is always connected to logic zero i.e. ground and hence is not brought out on TLL6219 platform.

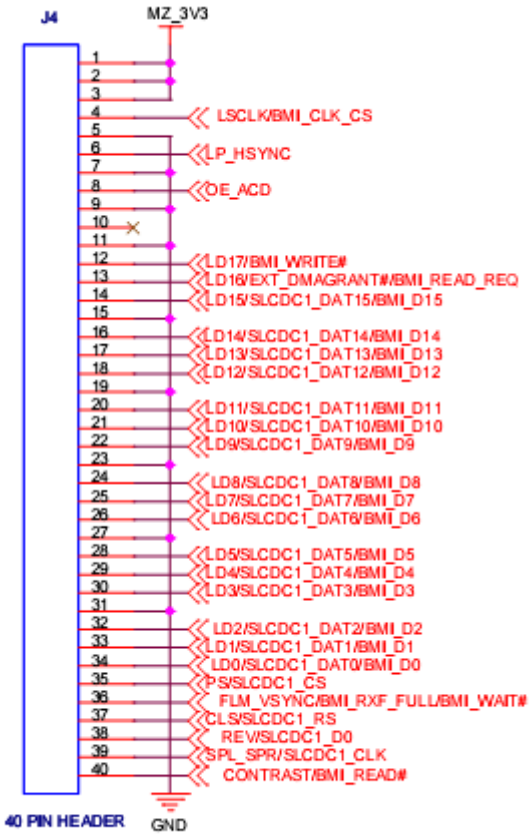
Refer GPCR (Global Peripheral Control Register) for Details on Boot options.



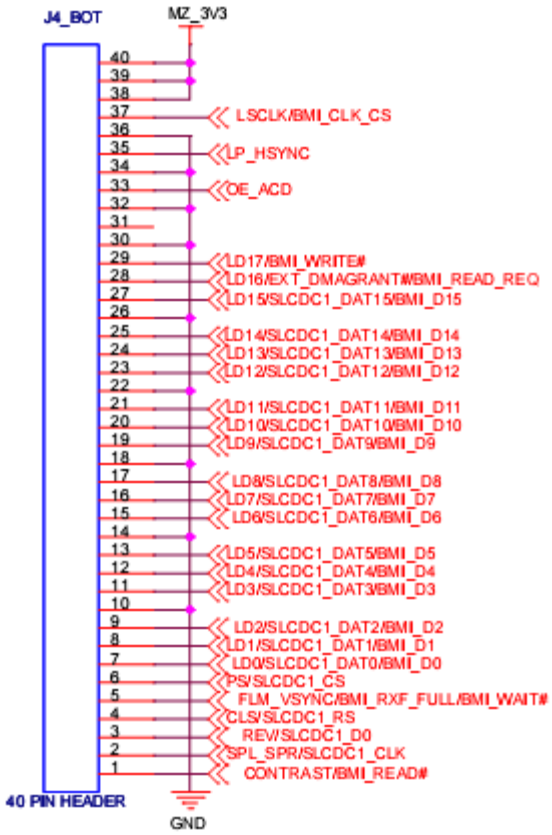
Inputs BOOT[3:0]	Output Signals Active Device	Boot Address
0000	iROM (Bootstrap USB/UART)	0x00000030
0001	iROM (Bootstrap USB/UART)	0x00000030
0010	iROM (8-bit 2 Kbyte NAND Flash)	0xDF003000
0011	iROM (16-bit 2 Kbyte NAND Flash)	0xDF003000
0100	iROM (16-bit 512 byte NAND Flash)	0xDF003000
0101	iROM (16-bit CS0 at D[15:0] (NOR Flash))	0xC8000000
0110	iROM (32-bit CS0 at D[31:0] (NOR Flash))	0xC8000000
0111	iROM (8-bit 512 byte NAND Flash)	0xDF003000

C.8 LCD Connector

TLL6219 provides two LCD connectors. LCD can be connected upwards or can be flipped downwards. The TLL6219 LCD connectors can interface with SLCD (smart LCD) also. Support for both monochrome and color modes. The SLCDC module has capability of directly interfacing with selected display devices.



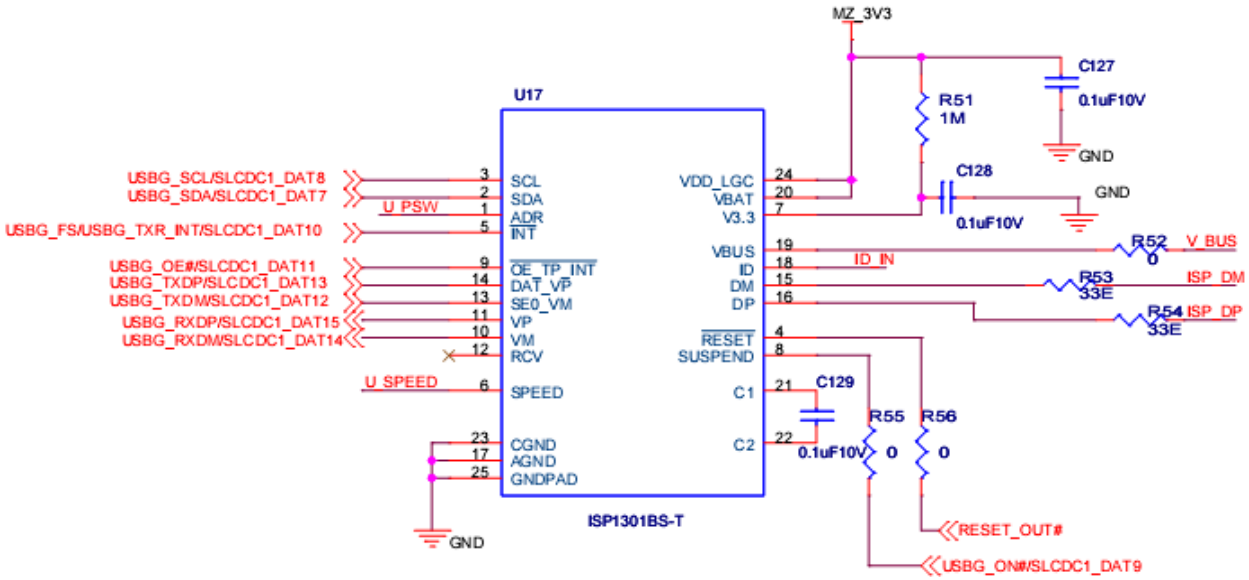
LCD CONNECTOR_TOP

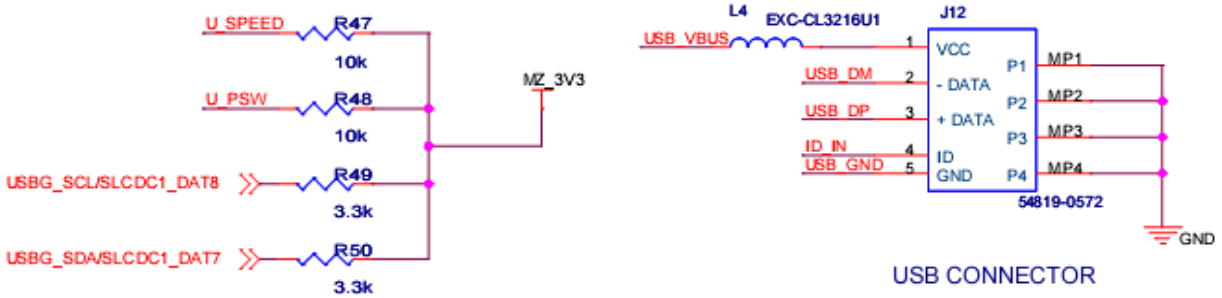


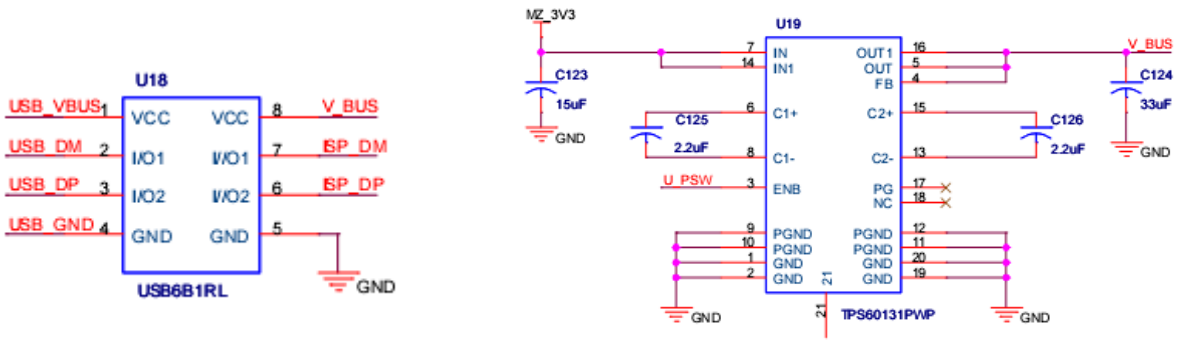
LCD CONNECTOR_BOT

C.9 USB

TLL 6219 platform has another form of communication i.e. Mini – USB which is a 5 pin connector. Between the i.MX21 and mini-USB is the OTG USB transceiver chip as shown in the schematic below.

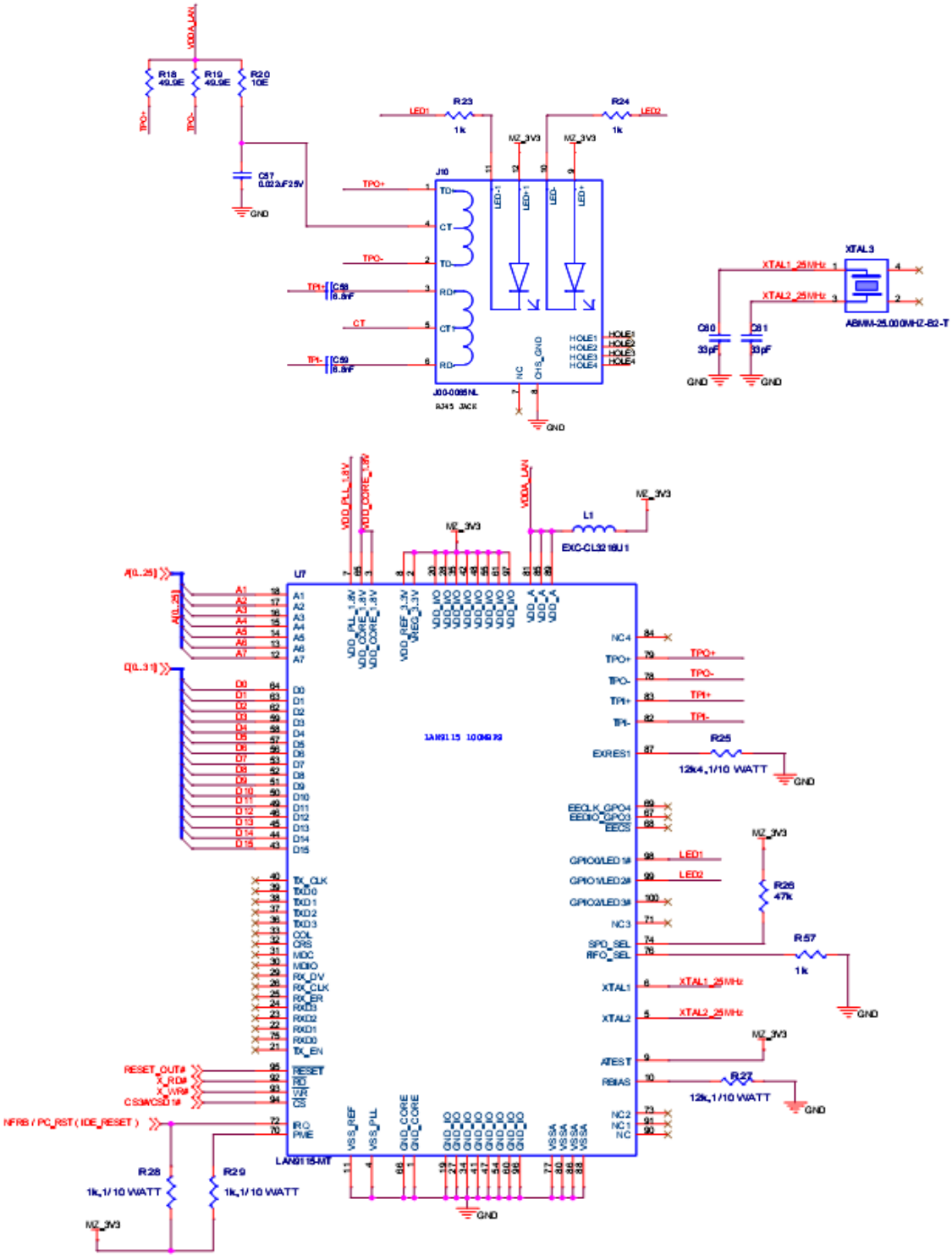






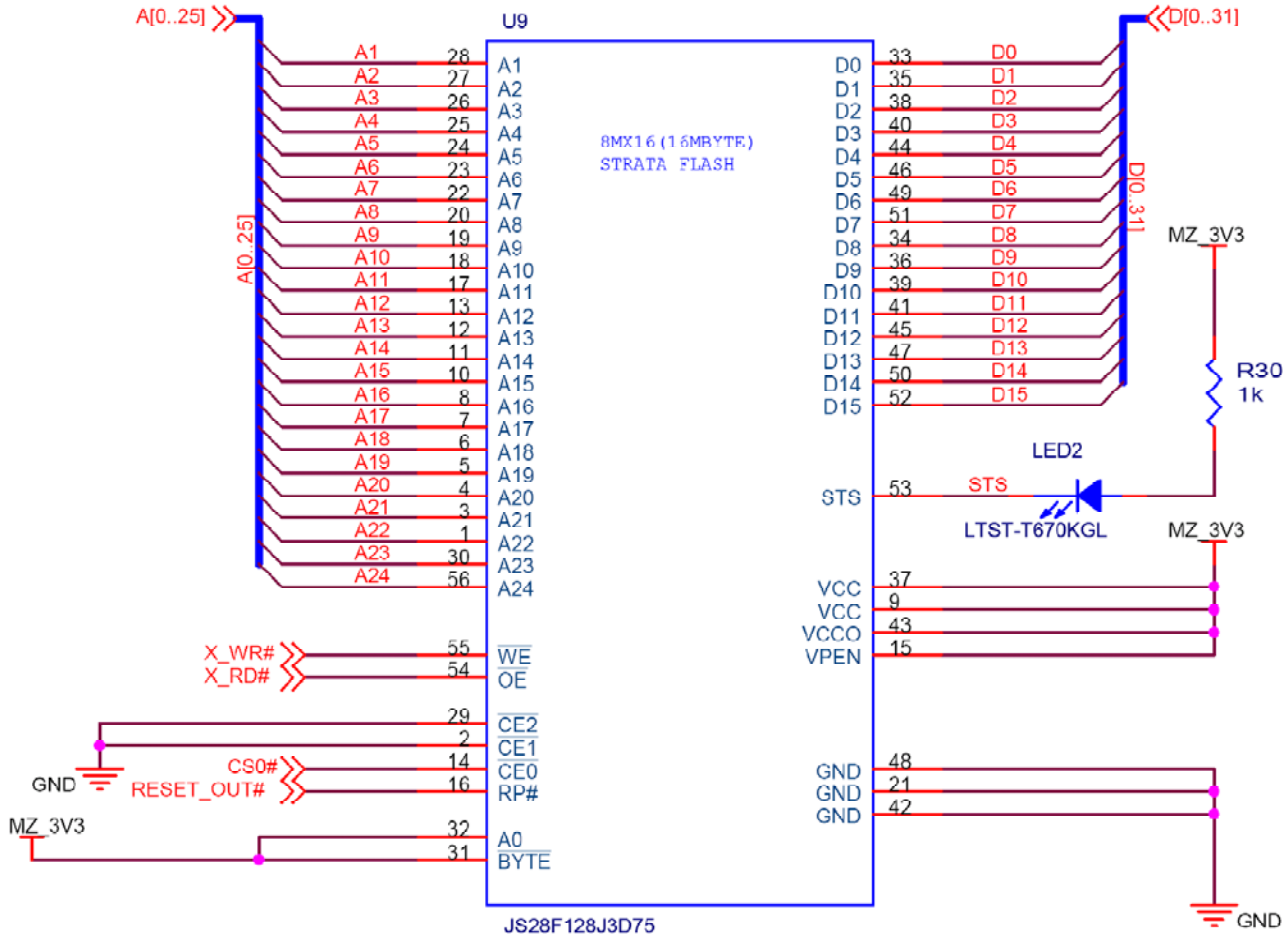
C.10 Ethernet

TLL SILC6219 platform provides a faster Ethernet is not fsater mode of communication i.e Ethernet (RJ45) connector. This can be used for TFTP (trivial file transfer protocol) with umon (Boot monitor) and linux.The highly efficient single chip LAN9115 SMSC Ethernet controller chip is used.



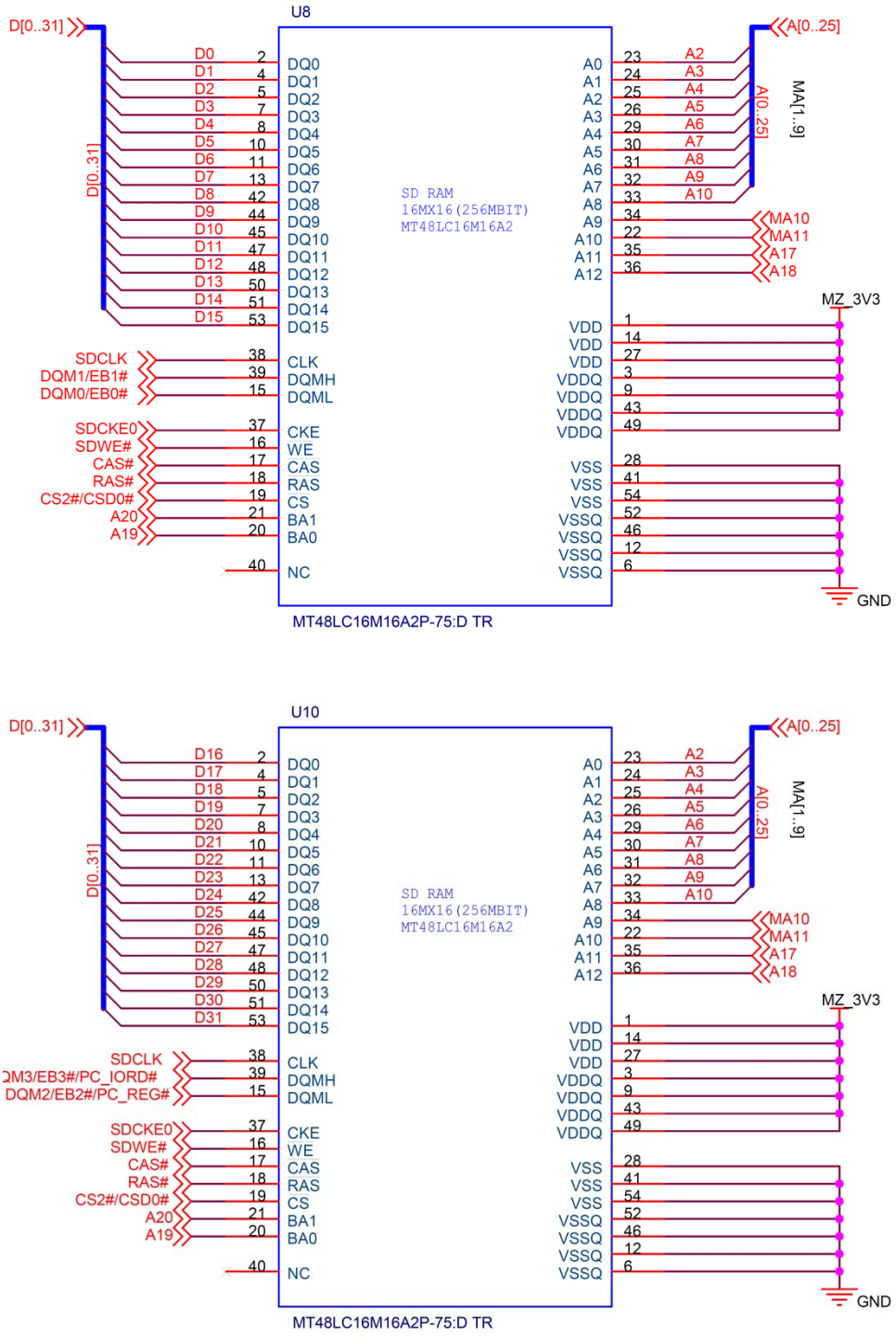
C.11 Flash Memory

The SILC-6219 utilizes a **16MB** of NOR Flash.

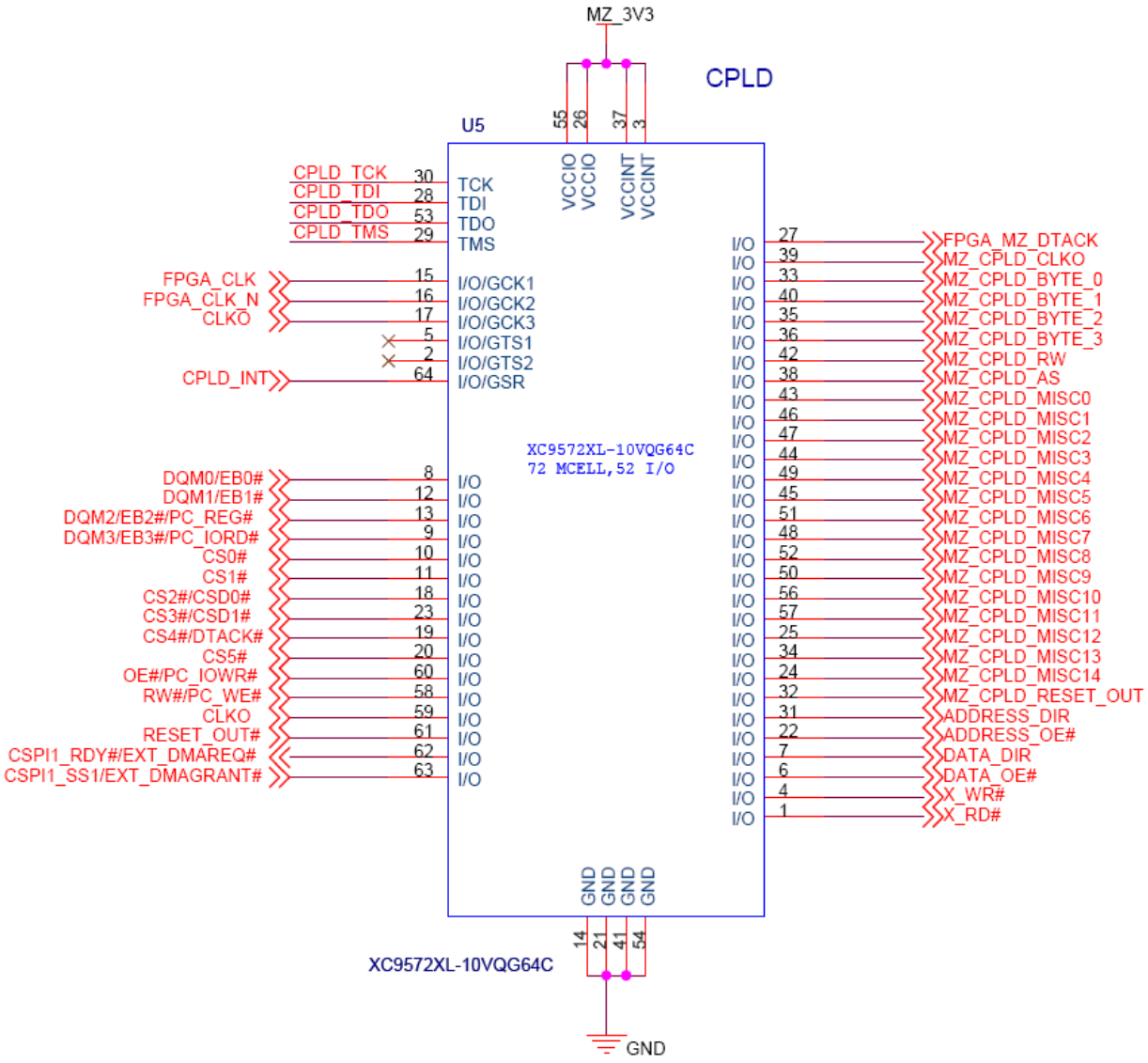


C.12 SDRAM

TLL SILC6219 Platform has two SDRAM chips each of 32MB. The total SDRAM memory is therefore **64MB**.

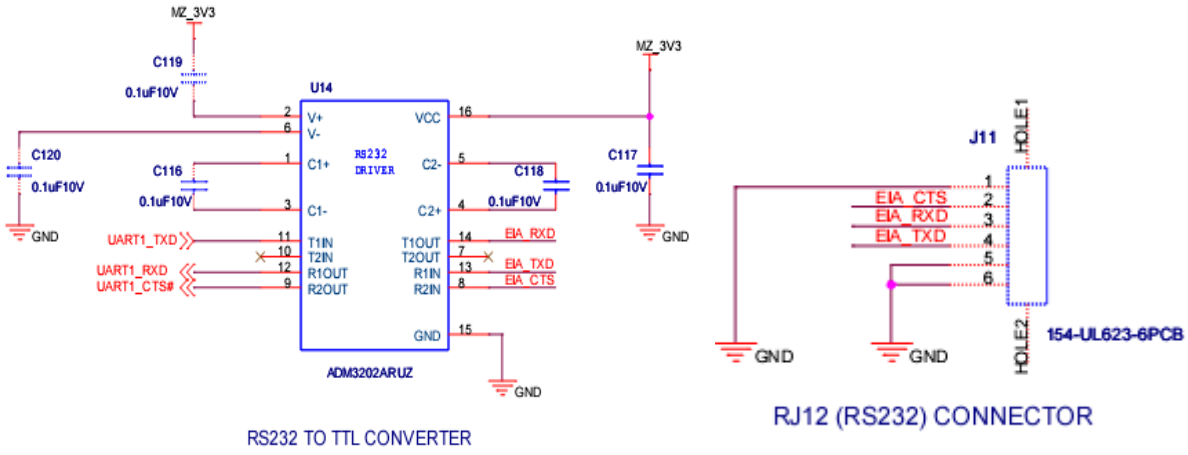


C.13 CPLD



C.14 Serial RJ12

TLL SILC6219 provides serial RJ12 port connected to UART1 of i.mx processor. The RJ12 is a standard telephone jack 6pin connector. The cable required for the serial communication is NULL modem cable (Tx → Rx, Rx→ Tx, Gnd → Gnd)



Frequently Asked Questions

- Why NOR Flash?
  - — NOR flash have fast Random reads and are best suited for code storage. Unlike NAND flash ,NOR flash memories are less prone to wear-leveling and possess a unique feature of XIP (eXecute In Place).NAND Flash are used mostly for mass storage.
- What is the Need of CPLD?
  - — CPLDs are Nonvolatile, Reprogrammable ,Low cost logic Devices.Programming them to build glue logic is very cost effective. Programming kept on power down, CPLD functions available instantly on system power up, almost impossible to steal stored design; Improves security, simplifies design. Their Role is also to glue together multiple devices which operate at different voltage levels.
- What is OTG?
  - — OTG stands for On The Go and is a term related to USB OTG and caters to the need of consumers to directly connect their OTG compatible devices to each other without any need of HOST PC. Unlike USB which requires a HOST and a peripheral for communication, in OTG Host and Peripherals can swap their roles. USB OTG devices are fully compatible with USB2.0. A typical example would be digital camera connecting to a printer.
- What is JTAG?
  - — JTAG stands for Joint Test Action Group. JTAG is a serial protocol, similar to SPI in some respects, that is used for boundary scan testing, in circuit emulation and flash programming.
- What is the need of IDE like eclipse?
  - — Eclipse provides a single user interface for editing the code, creating make files ,building projects , debugging and profiling and thus makes the job easy. All the above steps can be done separately from command line without using any IDE.
- What is SILC?
  - — SILC stands for Systematic Innovation and Learning Center. 6 is for pluggable mezzanine modules, 21 stands for i.mx21 soc, 9 stands for ARM9 core.