AIOT7-GMLi



Version 1.0

4.1 BIOS 24
4.2 BIOS 24
4.2.1 BIOS 24
4.2.2 24
4.3 Main 25
4.4 Advanced 26
4.5 Chipset 27
4.6 Security 28
4.7 Boot 29
4.8 Save & Exit 30
5 31
6 WDT 32
6.1 32
6.1.1 IO 32
6.2 WatchDog 33
6.2.1 WatchDog 33
6.2.2 33
7 GPIO 35
7.1 35
7.1.1 35
7.2 GPIO 37
7.2.1 37
7.2.2 38
1 39
2 40

1.3

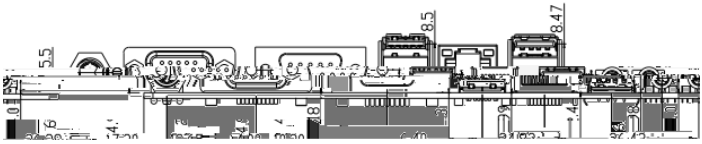
1.4 IO

DCIN)ˇ

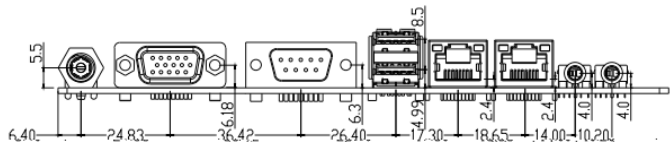
+HDMI



+COM



+COM

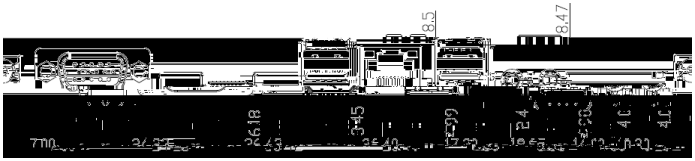


+HDMI

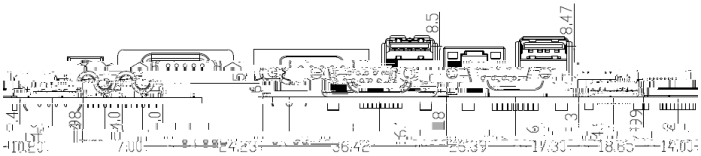


DCIN)

+HDMI

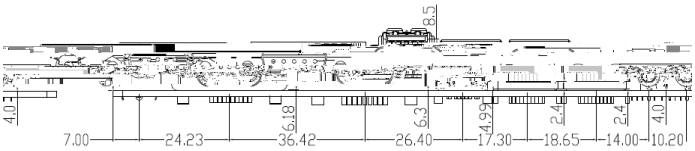


+COM

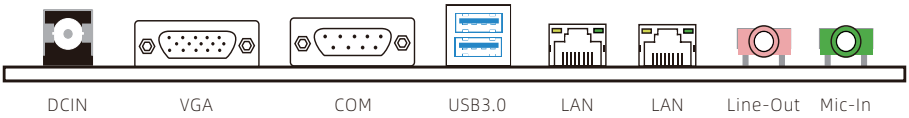


+HDMI

+COM

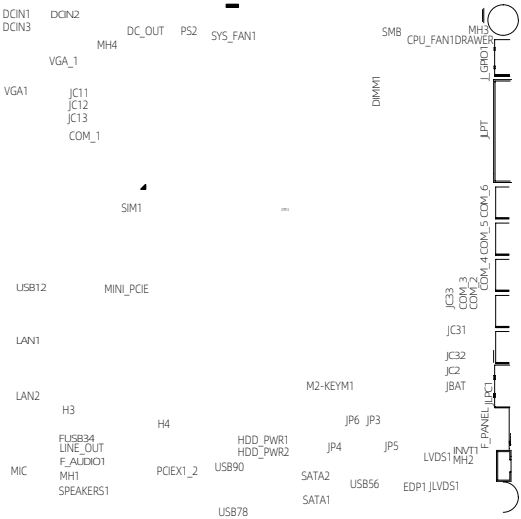


( )



( )

• DCIN: DC
• VGA: VGA
• COM:
• HDMI: HDMI
• USB3.0: USB3.0
• LAN: RJ45
• USB2.0: USB2.0
• LINE_OUT:
• MIC_IN:


DCIN1
DCIN3
DCIN2
DC_OUT
PS2
SYS_FAN1
MH4
VGA_1
VGA1
JC11
JC12
JC13
COM_1
SIM1
USB12
MINI_PCIE
LAN1
LAN2
H3
H4
FUSB34
LINE_OUT
F_AUDIO1
MH1
SPEAKERS1
MIC
PCIEX1_2
USB90
HDD_PWR1
HDD_PWR2
USB78
JP6
JP3
JP4
JP5
SATA2
SATA1
USB56
EDP1
JLVDS1
LVDS1
INVT1
MH2
M2-KEYM1
JC31
JC32
JC2
JBAT
JC33
JCOM_3
COM_2
COM_4
COM_5
COM_6
F_PANEL
JLPC1
JLPT
JGPIO1
DIMM1
SMB
CPU_FAN1
DRAWER
MH3

3 &

3.1
2

3.2

EDP LVDS LCD

1.
2. BIOS

JBAT

SHORT	CLEAR CMOS
OPEN	NORMAL

COM1 :
JC11/12/13

RS232	RS422	RS485
JC11(1-2)	JC11(5-6)	JC11(3-4)
JC12(1-3)	JC12(3-5)	JC12(3-5)
JC12(2-4)	JC12(4-6)	JC12(4-6)
JC13(1-3)	JC13(3-5)	
JC13(2-4)	JC13(4-6)	

COM2 :
JC2

1-2	5V
3-4	RI#
5-6	12V

COM3
JC32/33

TTL	RS232
JC32(3-4)	JC32(2-4)
JC32(7-8)	JC32(1-3)
JC33(1-3)	JC32(6-8)
JC33(2-4)	JC32(5-7)
	JC33(3-5)
	JC33(4-6)

JC31

1-2	5V
2-3	RI#
5-6	12V

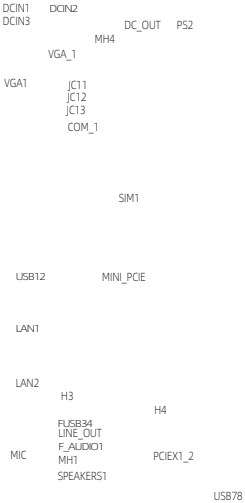
EDP/LVDS
JP3/4/5/6

LVDS	EDP
JP3(3-5)	JP3(1-3)
JP3(4-6)	JP3(2-4
JP4(3-5)	JP4(1-3)
JP4(4-6)	JP4(2-4
JP5(3-5)	JP5(1-3)
JP5(4-6)	JP5(2-4
JP6(3-5)	JP6(1-3)
JP6(4-6)	JP6(2-4

JLVDS (EDP LVDS LCDVCC)

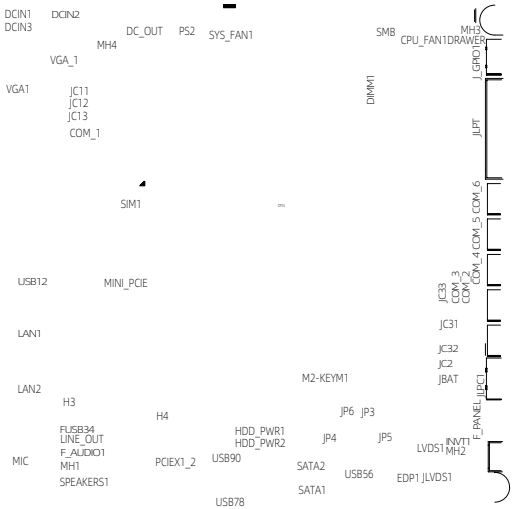
1-2	3.3V
2-3	5V

3.5 JGPIO



3.6 DRAWER

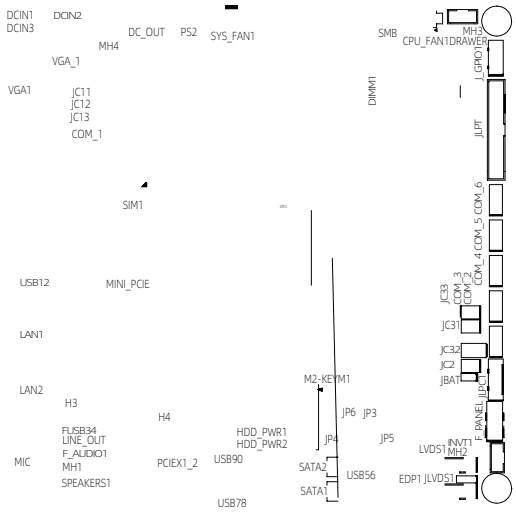
3.7 CPU_FAN

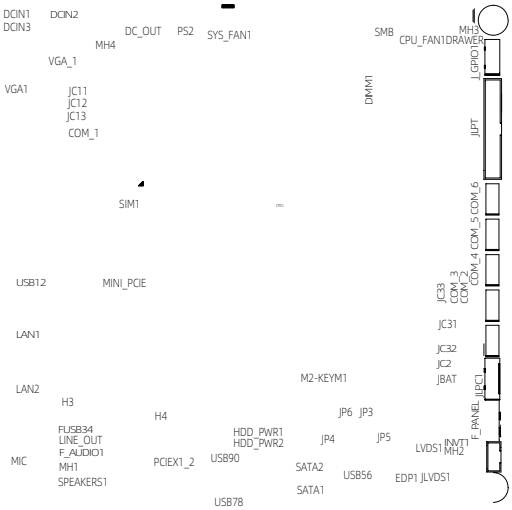


3.8 SMB SM Bus

3.11 DC_OUT


DCIN1
DCIN2
DCIN3
DC_OUT
MH4
VGA_1
VGA1
JC11
JC12
JC13
COM_1
USB12
LAN1
LAN2
H3
FUSB34
LINE_OUT
F_AUDIO1
MIC
MH1
SPEAKERS1


DCIN1
DCIN3
DCIN2
DC_OUT
PS2
SYS_FAN1
SMB
CPU_FAN1
MH3
DRAWER
MH4
VGA_1
J_GPIO
VGA1
JC11
JC12
JC13
COM_1
DIMM1
JLPT
SIM1
COM_6
COM_5
COM_4
USB12
MINI_PCIE
JC33
COM_3
COM_2
JC31
LAN1
JC32
JC2
JBAT
M2_KEYM1
LAN2
H3
H4
JP6
JP3
JP4
JP5
FUSB34
LINE_OUT
F_AUDIO1
MIC
MH1
SPEAKERS1
PCIEX1_2
HDD_PWR1
HDD_PWR2
USB90
SATA2
SATA1
USB56
USB78
LVDS1
INVT1
MH2
EDP1
JLVDS1
F_PANEL1


DCIN1
DCIN3
DCIN2
DC_OUT
PS2
SYS_FAN1
MH4
VGA_1
VGA1
JC11
JC12
JC13
COM_1
SIM1
USB12
MINI_PCIE
LAN1
LAN2
H3
H4
FUSB34
LINE_OUT
F_AUDIO1
MIC
MH1
SPEAKERS1
PCIEX1_2
USB90
HDD_PWR1
HDD_PWR2
SATA2
SATA1
USB56
USB78
JP6
JP3
JP4
JP5
M2-KEYM1
EDP1
JLVDS1
SMB
CPU_FAN1
DRAWER
MH3
DIMM1
J_GPIO1
JUPT
COM_6
COM_5
COM_4
COM_3
COM_2
JC33
JC31
JC32
JC2
JBAT
JLPC1
F_PANEL1
INVT1
MH2
LVDS1

DCIN1
DCIN3

VGA1

COM_ □

LAN1

LAN2

LINE_OUT
F_AUDIO1
MIC MH1

4 BIOS

4.1 BIOS

AMI BIOS BIOS Basic Input Output System(
ROM Read-Only Memory) BIOS

a. POST(Power On Self Test)
b.
c.
d. SETUP
BIOS CMOS RAM
BIOS CMOS
BIOS

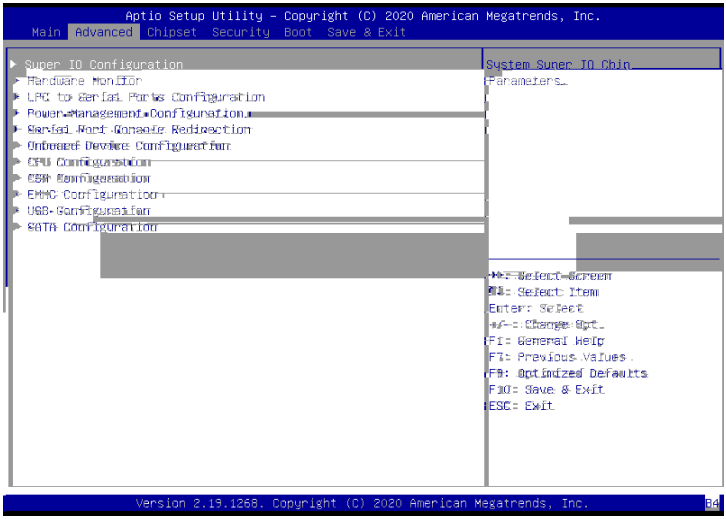
4.2 BIOS

BIOS Setup BIOS

SETUP

a. SETUP
b.

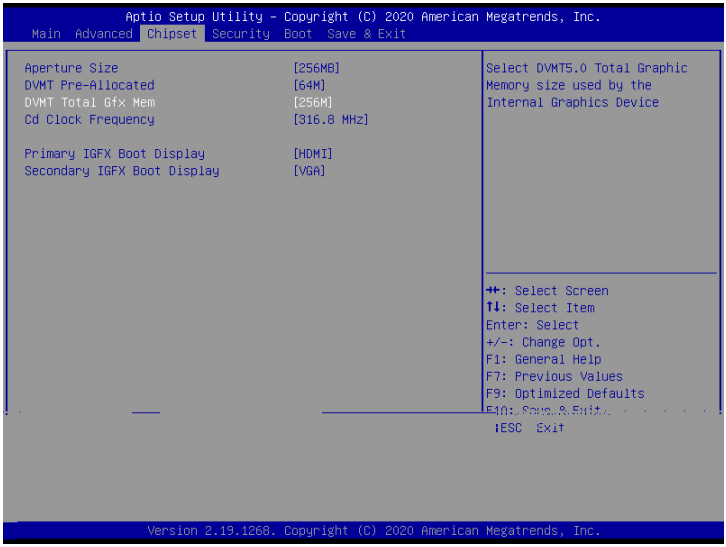
4.4 Advanced



Super IO Configuration

}

4.5 Chipset



Aperture Size

128M, 256M, 512M.

DVMT Pre-Allocated

DVMT5.0

64M, 96M, 128M, 160M, 192M, 224M, 256M, 288M, 320M, 352M, 384M, 416M, 448M,480M, 512M.

DVMT Total Gfx Men

DVMT5.0

128M, 256M, MAX.

Cd Clock Frequency

Cd

158.4MHz, 316.8MHz, 79.2MHz.

Primary IGFX Boot Display

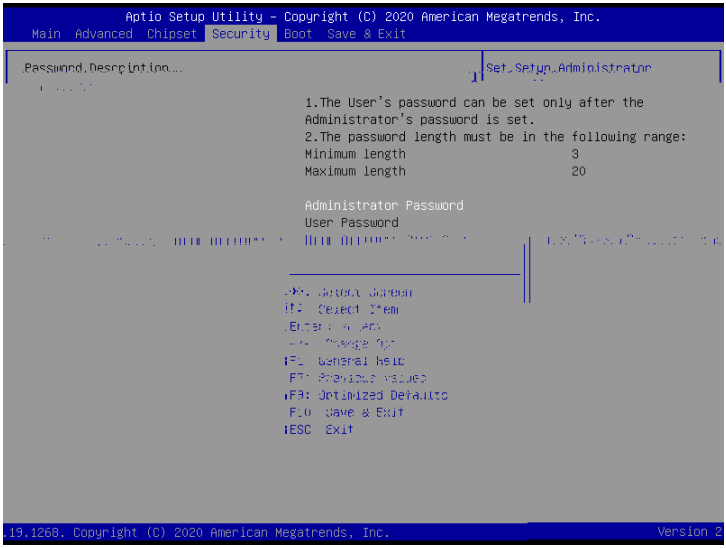
VGA, LVDS, HDMI.

Secondary IGFX Boot Disp

IGFX

Disabled, VGA, HDMI.

4.6 Security



Administrator Password

1. Administrator Password <Enter>

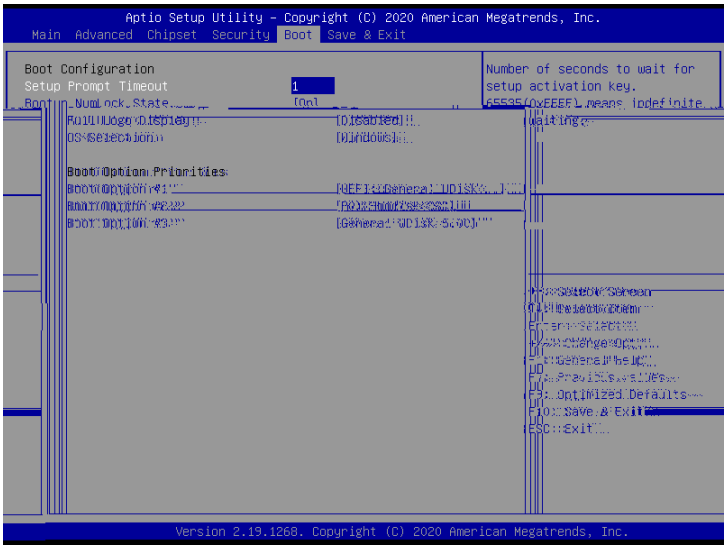
2. " Create New Password " 3~20 <Enter> " Confirm Password " " Invalid Password! "

" Administrator Password " " Enter Current Password " " Create New Password " <Enter>

User Password Boot Prompt

Enabled,Disabled.

4.7 Boot



Setup Prompt Timeout

Bootup Numlock State

Numlock On NumLock

Of Numlock

On,Of.

Full Loge Display

Enabled Disabled.

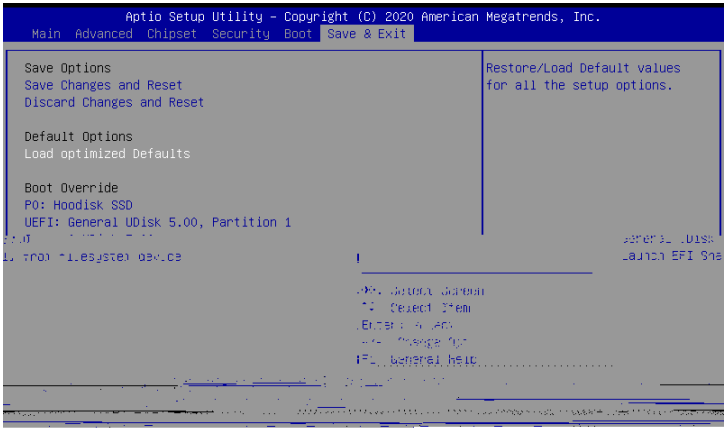
OS Selection

Windows, Android, MSDDS.

Boot Option #1/2/3

: PO: Hoodisk SSD, UEFI General UDisk 5.00,Partition 1, General UDisk 5.00, Disabled.

4.8 Save & Exit



Save Changes and Reset

Discard Changes and Reset

Load optimized Default

/

Launch EFI Shell from filesystem device

Shellx64.efi

5

X \AUTORUN.EXE X



( )

6 WDT

6.1

AIoT0-H110 WatchDog
WDT ITE8786 Super IO

6.1.1 IO

Linux

```
#include <stdio.h>
#include <errno.h>
#include <sys/io.h>

// IO 1 Byte
  /*
    Name:IoRead8
    Input: GpioAddress - GPIO base address
    Output: GPIO port read value
    Description:
  */
unsigned char IoRead8(unsigned short GpioAddress) {
  unsigned char ret;
  ret = iopl(3);
  if (ret)
  {
        printf("ret = %d\n", ret);
        printf("errno = %d\n", errno);
        printf("error for iopl\n");
        return 1;
      }
  return inb(GpioAddress);
// IO 1 Byte
unsigned char IoWrite8(unsigned short GpioAddress, unsigned char Data) {
  unsigned char ret;
  ret = iopl(3);
  if (ret)
  {
        printf("ret = %d\n", ret);
        printf("errno = %d\n", errno);
        printf("error for iopl\n");
        return 1;
  }
      outb(Data, GpioAddress);
      return 0;
  }
```

7 GPIO

7.1

AIoT7-GML GPIO
GPIO Intel PCH GPIO
GPIO

7.1.1 IO BIOS

```
/**
  Reads a 32-bit MMIO register.

  Reads the 32-bit MMIO register specifed by Address. The 32-bit read
value is
  returned. This function must guarantee that all MMIO read and write
  operations are serialized.

  If Address is not aligned on a 32-bit boundary, then ASSERT().

  If 32-bit MMIO register operations are not supported, then ASSERT().

  @param  Address The MMIO register to read.

  @return The value read.

**/
UINT32
EFIAPI
MmioRead32 (
  IN      UINTN                     Address
  )
{
  //
  // Make sure Address is aligned on a 32-bit boundary.
  //
  ASSERT ((Address & 3) == 0);
  return (UINT32)MmioReadWorker (Address, EfCpuIoWidthUint32);
}

/**
  Writes a 32-bit MMIO register.
```

```
  IN      UINTN                     Address,
  IN      EFI_CPU_IO_PROTOCOL_WIDTH  Width
  )
{
  EFI_STATUS  Status;
  UINT64      Data;
  Status = mCpuIo->Mem.Read (mCpuIo, Width, Address, 1, &Data);
  ASSERT_EFI_ERROR (Status);

  return Data;
}
/**
  Writes memory-mapped registers in the EFI system memory space.

  Writes the MMIO registers specified by Address with registers width and
value specified by Width
  and Data respectively. Data is returned. If such operations are not
supported,then ASSERT().
  This function must guarantee that all MMIO read and write operations
are serialized.

  @param  Address      The MMIO register to read.
                  The caller is responsible for aligning the Address
if required.
  @param  Width        The width of the I/O operation.
  @param  Data         The value to write to the I/O port.

  @return Data read from registers in the EFI system memory space.

**/
UINT64
EFIAPI
MmioWriteWorker (
  IN      UINTN                     Address,
  IN      EFI_CPU_IO_PROTOCOL_WIDTH  Width,
  IN      UINT64                    Data
  )
{
  EFI_STATUS  Status;

  Status = mCpuIo->Mem.Write (mCpuIo, Width, Address, 1, &Data);
  ASSERT_EFI_ERROR (Status);

  return Data;
  }
```

7.2 GIOP

7.2.1 8 GPIO

GPIO1 0xD0C50A30
GPIO2 0xD0C50A40
GPIO3 0xD0C50A50
GPIO4 0xD0C50A60
GPIO5 0xD0C509A0
GPIO6 0xD0C509B0
GPIO7 0xD0C509C0
GPIO8 0xD0C509D0

					USB3	USB2	COM	LAN	SSD	PCIe
AIoT7-GMLi1	J4025	1	3	2SATA	2	8	6	1 x i211	M.2	2
AIoT7-GMLi2	J4025	1	3	2SATA	2	8	6	2 x i211	M.2	2
AIoT7-GMLi3	J4105	1	3	2SATA	2	8	6	2 x i211	M.2	2
AIoT7-GMLi4	J4125	1	3	2SATA	2	8	6	2 x i211	M.2	2

