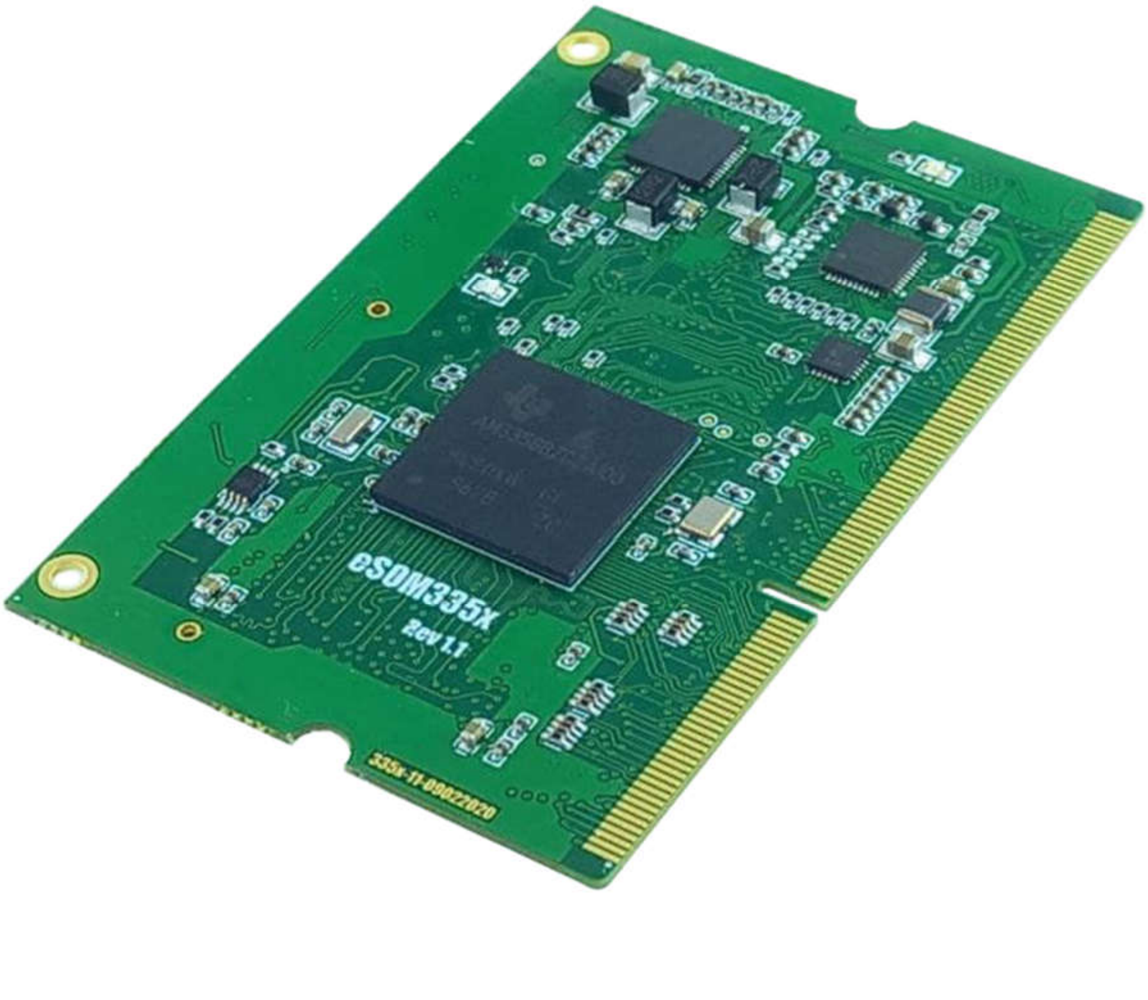
eSOM335x

Quick Start Guide



Contents

Introduction

The eSOM335x is a small form factor System on Module(SoM) based on AM3358BZCZA100 ARM Cortex-A8 microprocessor family from Texas Instrument. The AM335x microprocessors are enhanced with image, graphics processing, peripherals and industrial interface options. The PowerVR SGX™ Graphics Accelerator subsystem provides 3D graphics acceleration to support display and gaming effects. The PRU-ICSS enables additional peripheral interfaces and real-time protocols such as EtherCAT, PROFINET, EtherNet/IP, PROFIBUS, Ethernet Powerlink, Sercos, and others.

The eSOM335x was designed very flexible, its memories are replaceable easily without any soldering work. RAM and eMMC units were designed modular and plug into coreboard through two 60pin specific connectors. DDR3L RAM modules come with 512MB capacity in standard version, but customizable to 128, 256, 1024MB depend upon customer needs. The eMMC modules exist in 8GB with 8bit Bus width by default, and it is customizable with 2, 4, 16, 32GB versions. Upgrade, repair and troubleshooting in this way is easy, quick, and cost-effective and improve maintenance ability.

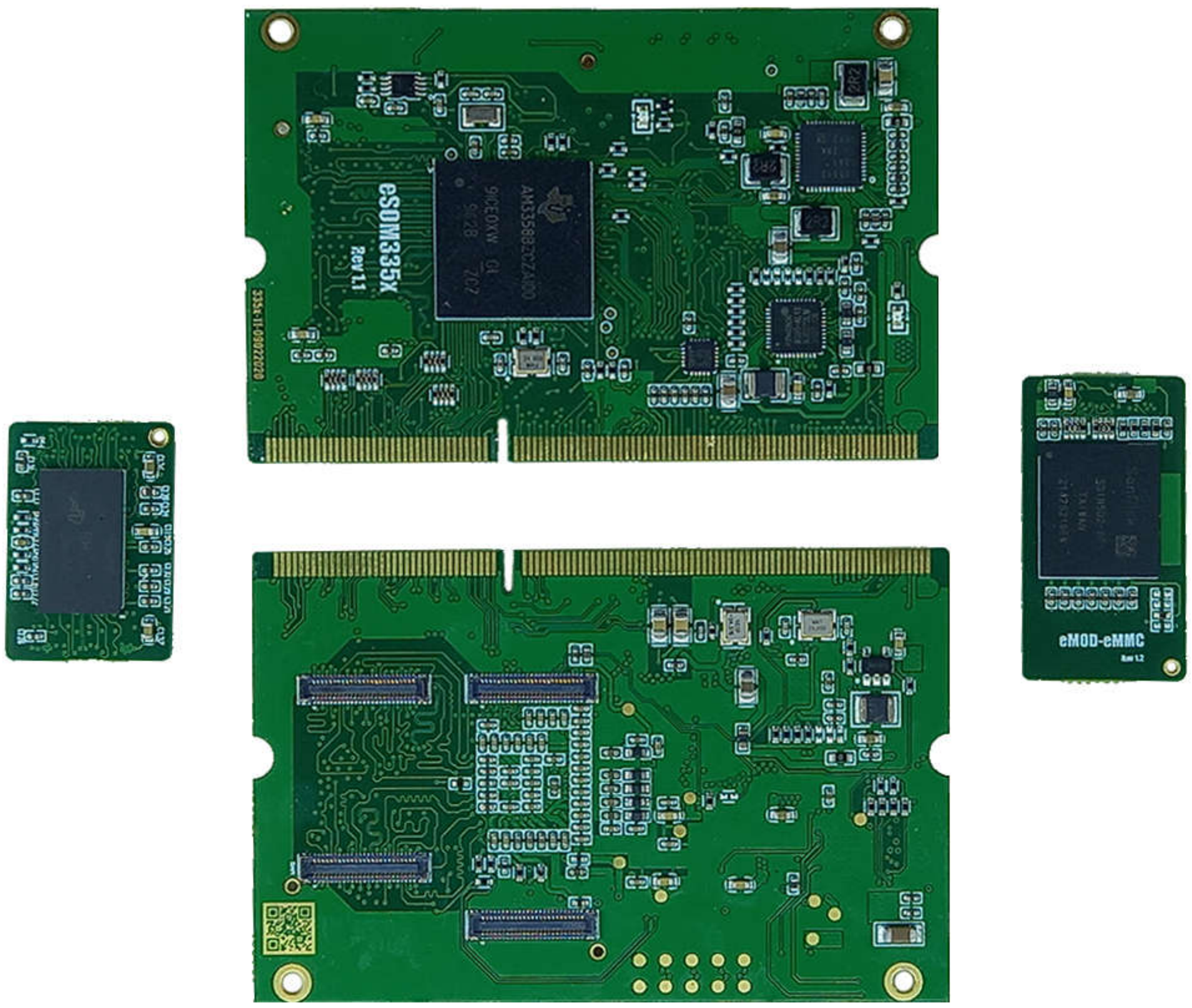
This SOM with small footprint 67.6mm × 44.0mm × 5.6mm allows designers to leverage the ARM hardware and WinCE software stack in their own custom systems. The eSOM335x integrated a PMU to reduces complexity of external power supply. This module has extra IO interfaces like HDMI, LCD, Ethernet, USB, GPIO, UART, I2C, ADC and etc, that users can access them through DDR3-SODIMM 204 pin straight connector that enables hardware customization and gives more hardware flexibility to various projects.

The eSOM335x can boot a real-time, small foot-print Windows CE very quickly. OS images are customized to enable users on rapid application development without any OS challenging. Users are free to choose the video output and can change their LCD without need to OS customization. In addition, users can configure their boot setting such as their logo and boot progress bar easily. Many device drivers like 4G module, Camera, CAN, Touch are integrated in OS image and users can add them to their projects safely. In addition, many sample codes are provided for software developer to access to eSOM335x peripherals and accelerates time-to-market procedures.

This tiny single board computer targets a wide range of applications, including: Medical Devices, PLC, Industrial Automation, HMIs, Entertainment system, POS, Data Acquisition, Gaming and much more.

Features

- Texas Instrument AM3358BZCZA100 ARM Cortex-A8 Up to 1GHz Processor
- Replaceable and optional modular memories
- Support HDMI, TFT and LVDS as Video Output
- Rich interface through DDR3-SODIMM 204 pin connector
- On-board Gigabit Ethernet PHY, customizable Up to 2 Ethernet port
- Support 4G Module
- Support Camera
- Ready to Run WIN CE7 with quick launching
- Easy configuration OS parameters and driver settings



Specification

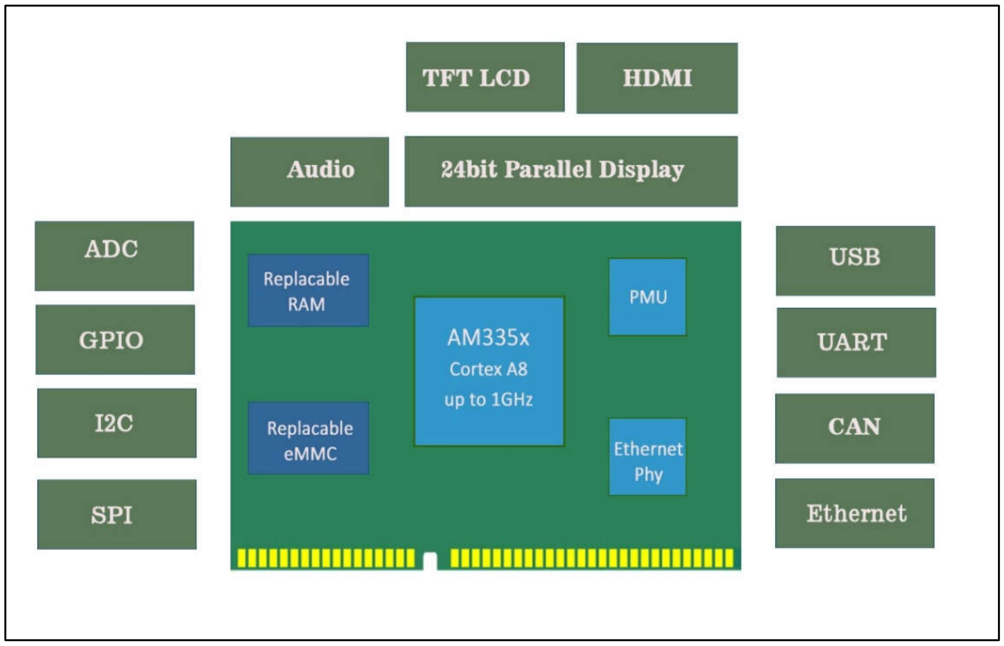
Characteristic	Standard	Customizable
Processor Details		
Processor	Texas Instrument AM3358BZCZA100 Up to 1GHz Sitara™ ARM® Cortex®-A8 32-Bit RISC Processor	AM3359 / AM3354 / AM3352
Memory		
RAM	512MB DDR3L 16bit BUS Replaceable module	128, 256, 1024MB
Storage	8GB eMMC 4.3 + ECC 8bit BUS Replaceable module	2, 4, 16, 32GB
Multimedia		
Display Type	Single Channel, 24bit Parallel Data Output	
Color Depth	24Bit	
Resolution	1024 x 768 (Tested)	Up to 2048 * 2048
Graphics Engine	PowerVR SGX530 3D	
Multichannel Audio Serial Ports	-	2

Analog Sound (Over MASP0)	HeadPhone(Stereo)/ MIC(Mono)
	LineOut(Stereo) (not implemented on OS)
	LineIn(Stereo) (not implemented on OS)	

Connectivity		
USB	2 HOST Mode	Dual OTG Function
UART	5 (RX,TX)	6 (RX, TX, RTS, CTS)
SPI	1	2
I2C	2	3
Ethernet	1 (10/100/1000)	2 (10/100/1000)
CAN	1	2
SDIO	1 (4bit SD-Card Reader)	1 (8bit SDIO) + 1(4bit SD-Card Reader)
Analog Input	8 (12bit 200KS/s)	
GPIO	17	77
OS		
WinCE	CE 7	Debian
Physical		
Size	67.6 x 44.0 x 5.6	
Working Temperature	0 to +70	
Power	5V 700mA (Max)	

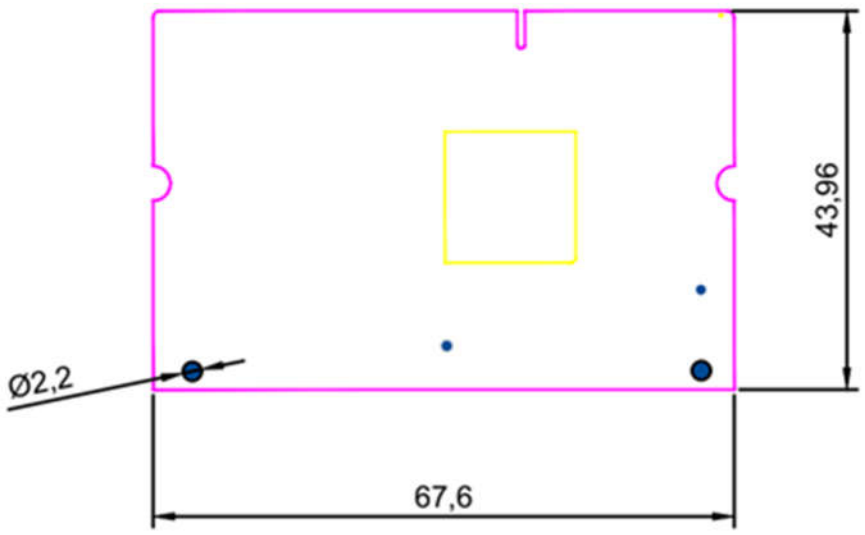
Functional Block Diagram

Functional block diagram of eSOM335x is described in following picture, for pin assignment information refer to ***pinMux*** document.



Mechanical Drawing

Dimension	
PCB	6 Layer
Weight	11g
Size	67.6mm × 43.96mm × 5.6mm



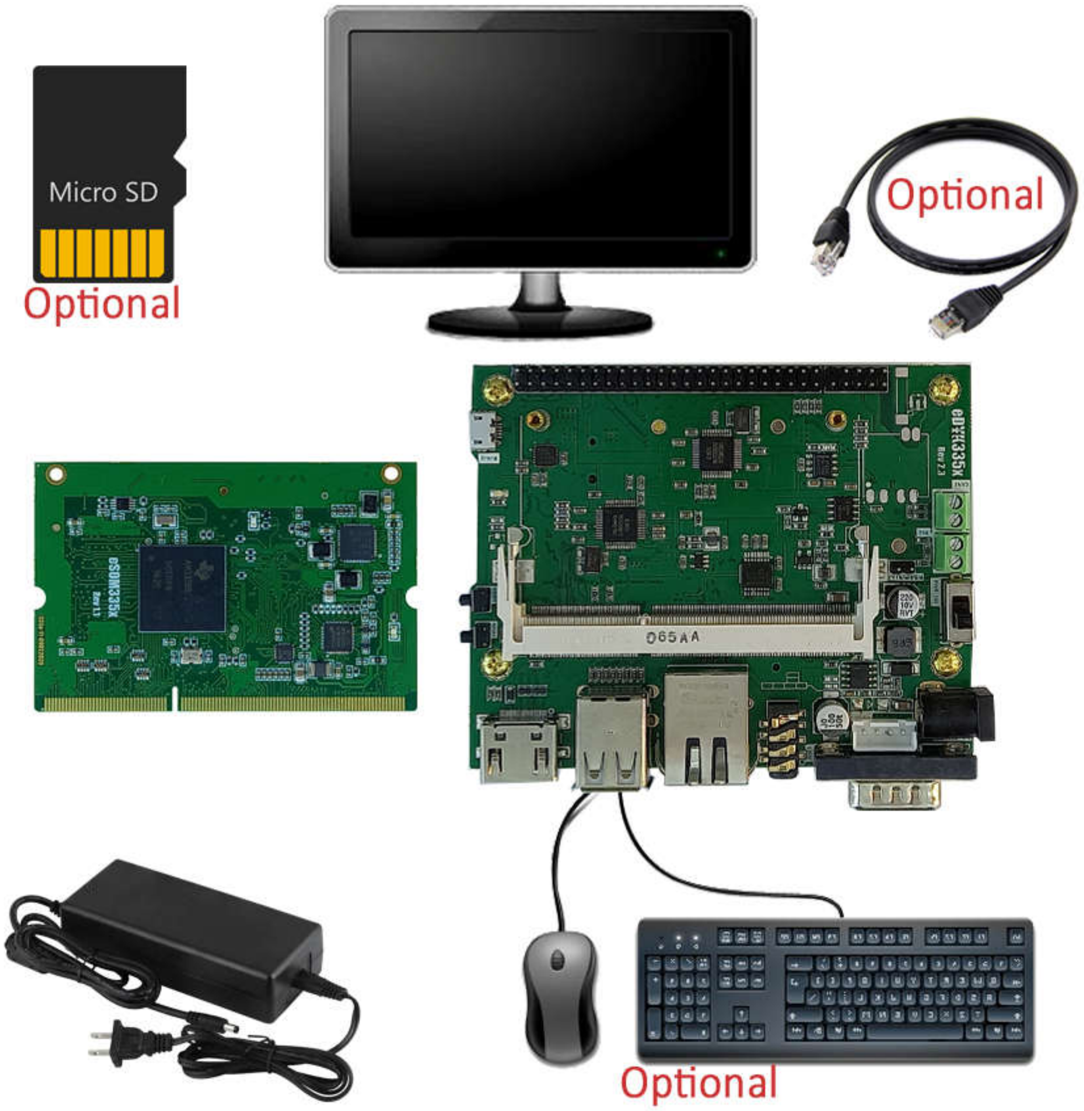
Power ON

Essentials You Need

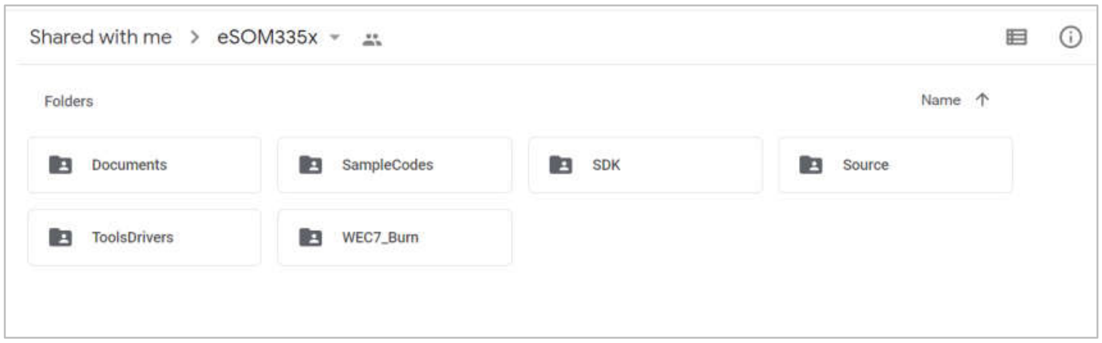
Users need to following items to power on eSOM335x:

- eSOM335x coreboard and development board (eDVK335x is recommended)
- HDMI display or LCD
- Proper power adaptor(if using eDVK335x proper power supply is 7.5V to 24V DC)
- Ethernet cable
- Micro USB cable
- USB mouse and keyboard
- microSD card

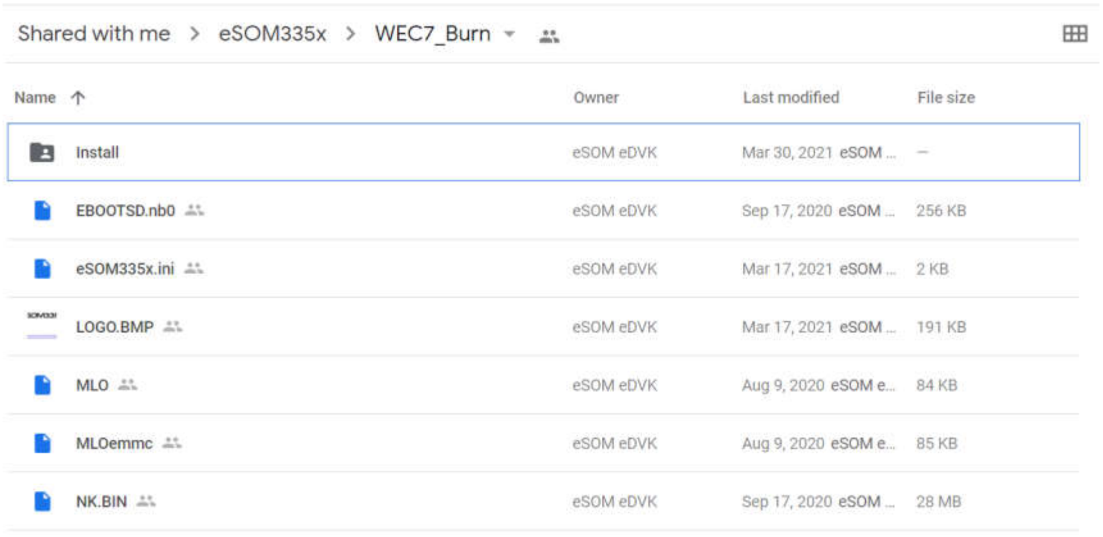
Essentials You Need



The eSOM335x come with many resources consist of OS images and sources, sample codes and documents that users can access them through a net drive URL after buy eSOM335x, following figure shows these files:

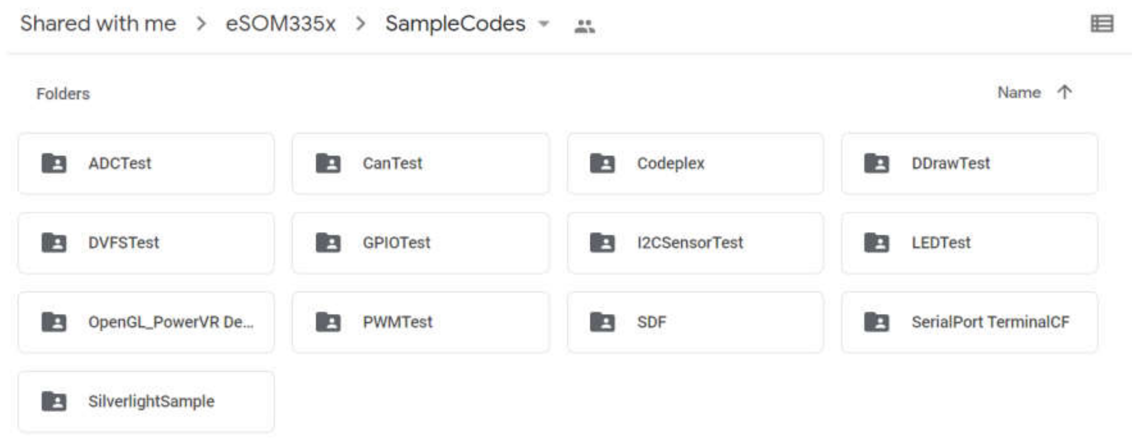


OS images exist in WEC3_Burn folder:


Shared with me > eSOM335x > WEC7_Burn

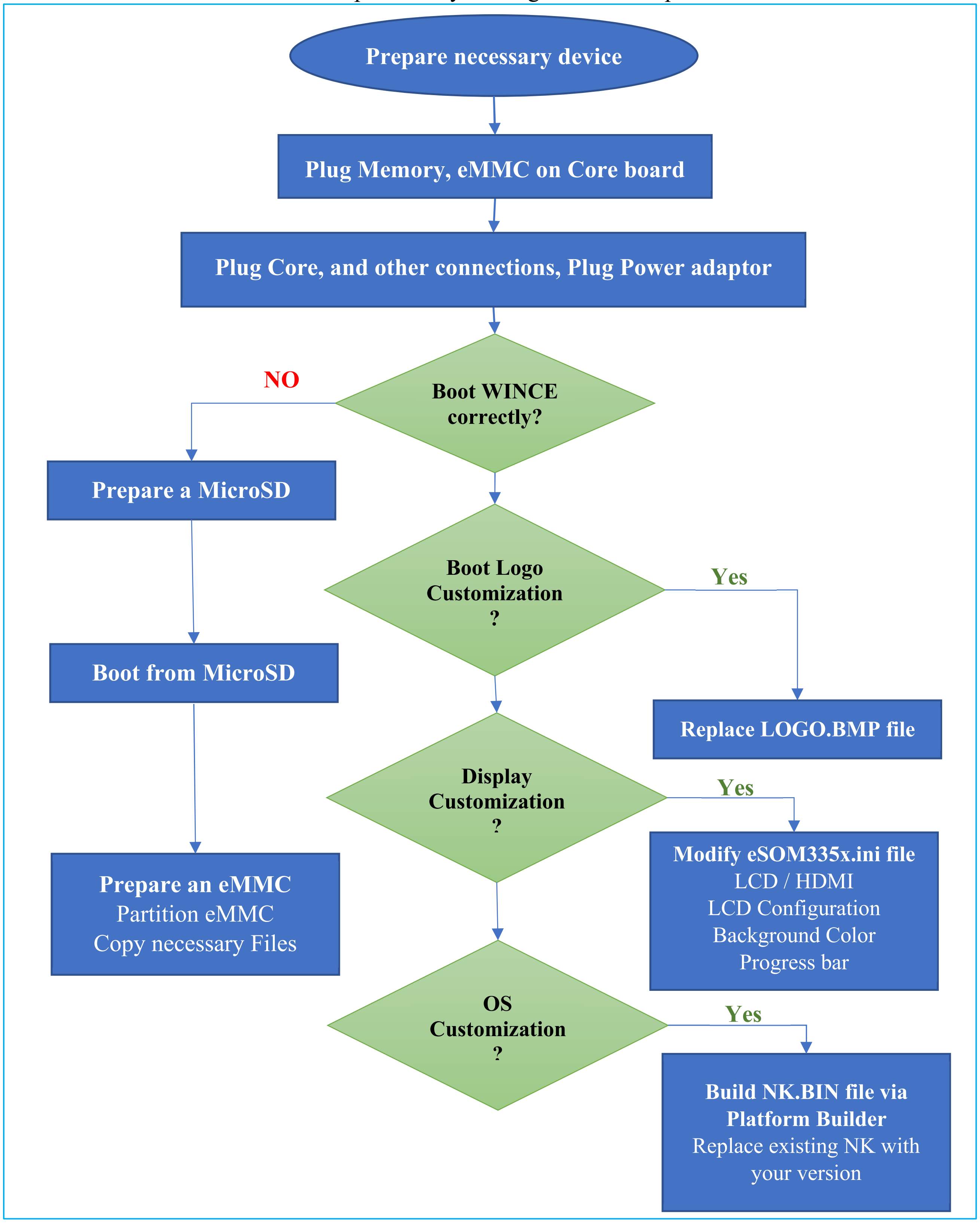
Name ↑	Owner	Last modified	File size
Install	eSOM eDVK	Mar 30, 2021 eSOM ...	—
EBOOTSD.nb0	eSOM eDVK	Sep 17, 2020 eSOM ...	256 KB
eSOM335x.ini	eSOM eDVK	Mar 17, 2021 eSOM ...	2 KB
LOGO.BMP	eSOM eDVK	Mar 17, 2021 eSOM ...	191 KB
MLO	eSOM eDVK	Aug 9, 2020 eSOM e...	84 KB
MLOemmc	eSOM eDVK	Aug 9, 2020 eSOM e...	85 KB
NK.BIN	eSOM eDVK	Sep 17, 2020 eSOM ...	28 MB

Many sample codes are provided for rapid programming as shown in the following figure:



eSOM335x Set up

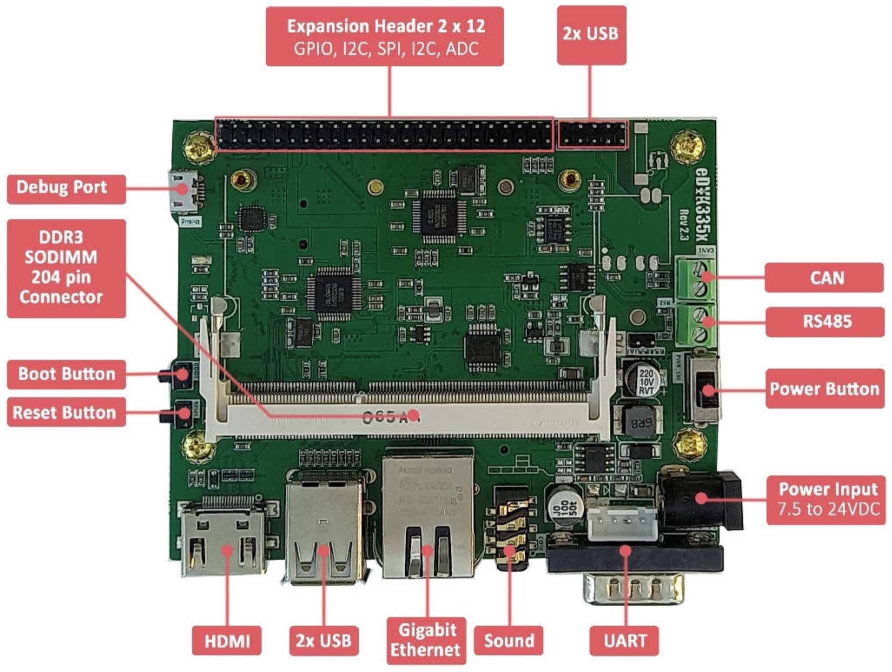
Follow these instructions to power on your single board computer:



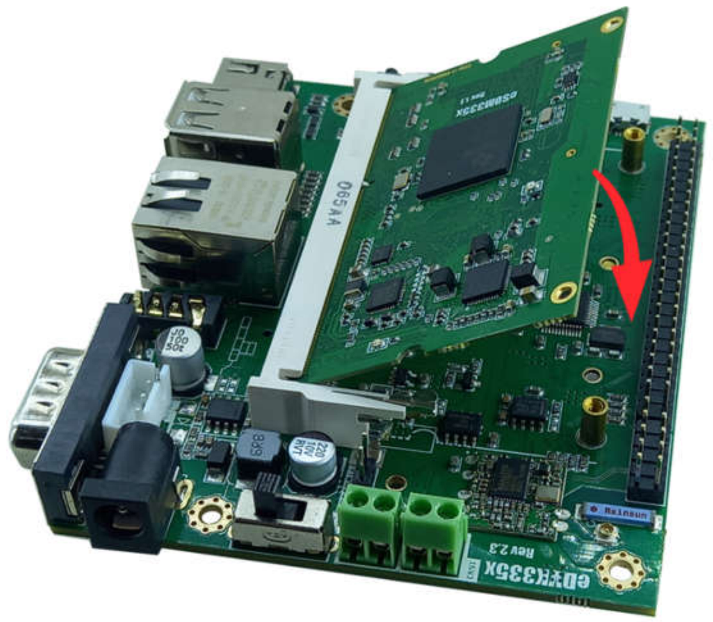
- Plug memory module into your eSOM335x memory connector:

- Plug eMMC module into your eSOM335x eMMC connector:

- For using eSOM335x you need to design or buy a development board, eDVK335x is a recommended development board for this coreboard as shown in the following figure:

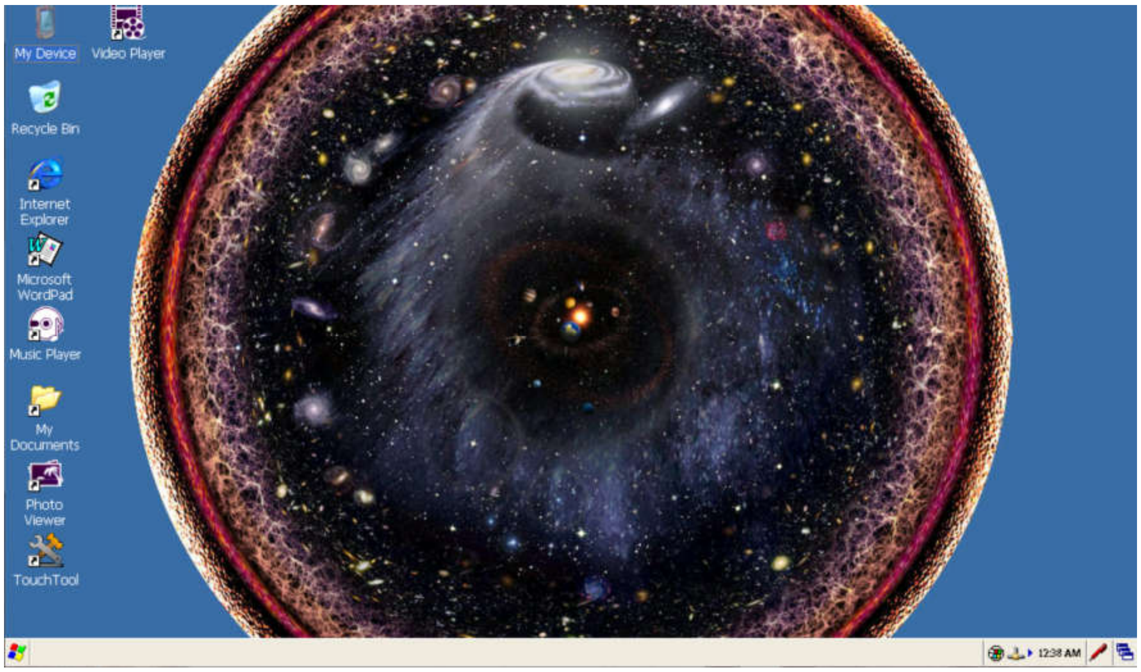


- Plug coreboard into development board corresponding slot:

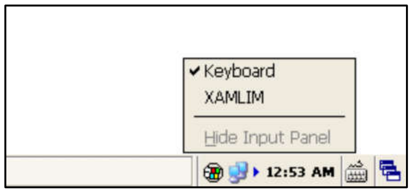


- Plug USB mouse and keyboard in USB port of your development board. In eDVK335x exist a USB stacked with 2 output port and 9 pin headers that user can use them for connecting USB devices like mouse, keyboard, camera etc.
- Plug Ethernet cable into your development board and Host PC. Ethernet connection is useful for deploy application on eSOM335x device from PC and remote desktop applications etc.
- eSOM335x support Single Channel, 24bit Parallel Data Output, and in eDVK335x a HDMI chip in integrated onboard that convert LCD data to HDMI interface. Users can connect HDMI display to HDMI port. In addition, a FPC connector is also used for connecting to TFT LCDs. users are free to choose LCD display. For using their LCD with eSOM335x needs to edit eSOM335x.ini configurations parameters explained in ***Boot Customization*** section.
- Users need to power the developer kit with a good quality power supply, (if use eDVK335x your power adaptor should deliver 7.5 to 24V with 2A current) at the developer kit's power jack. Plug Power adaptor into your development board.
- The eSOM335x can run WINCE OS from uSD and eMMC that required files prepared for them separately. After board power on, CPU check boot mode and initialized with corresponding mode. By default, eSOM335x uses an eMMC chip as a boot device and main storage. The eSom335x is preinstalled with WINCE on its eMMC, if for some reason eMMC module is corrupted or you want a new eMMC module with your requirements refer to ***Prepare eMMC*** section.

- After Power on your board WinCE launch and desktop appears:



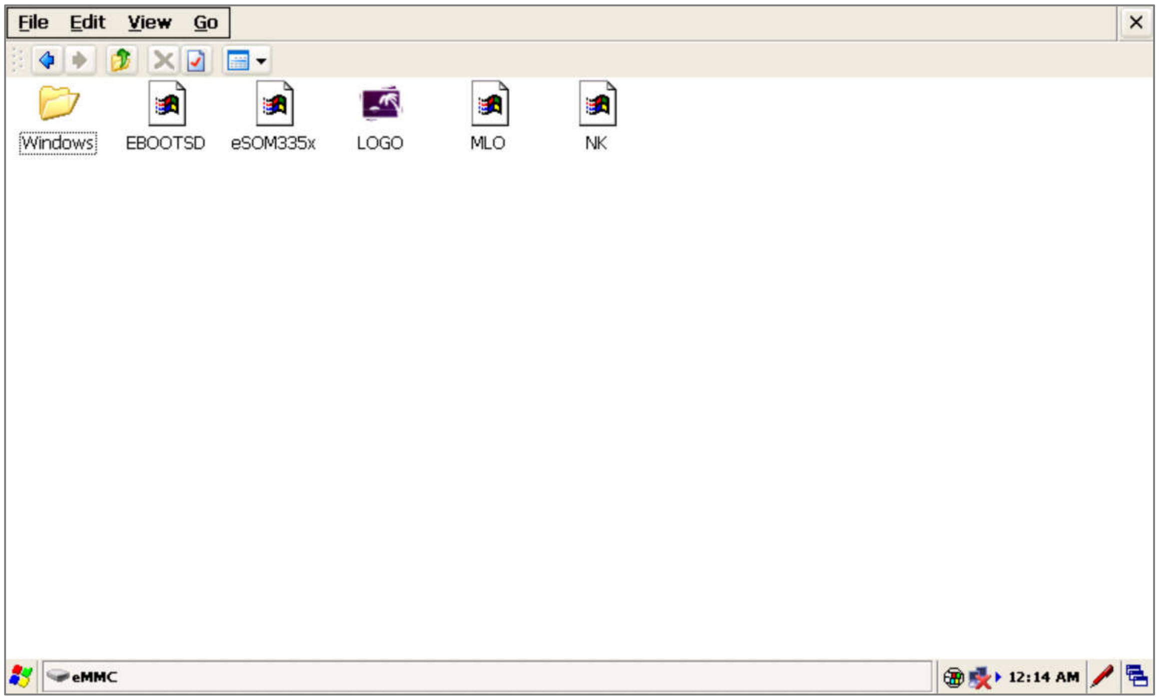
Two type of virtual keyboard are provided for user inputs that can selected from associated icon on taskbar as shown in below figure:



Boot Customization

The eSOM335x has options that users can edit them for their requirements, screen selection, boot logo and ProgressBar are customizable in this manner:

- After OS launched in your device navigate to eMMC drive, you can see OS files as shown in following picture:



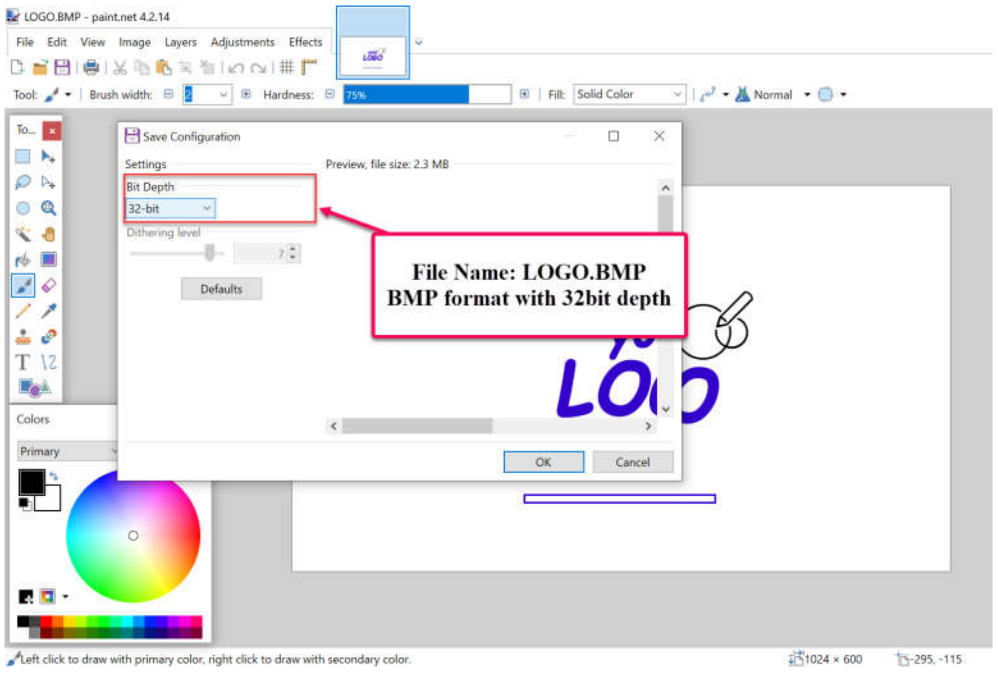
Following table lists these files:

File Name	Description
EBOOTSD.nb0	WinCE bootloader close source
eSOM335x.ini	Config file for Display and startup logo setting
LOGO.BMP	Picture with bitmap32 format for boot image
MLO	Initialize Hardware-close source
NK.BIN	WinCE kernel- open source

- **MLO:** Basically, MLO will initialize the strict minimum on the CPU to be able to copy the EBOOTSD.nb0 to external RAM.

- **EBOOTSD**: is the standard bootloader of Windows CE. It initializes much more peripherals and configure the boot settings. Basically, eboot will load NK.BIN to RAM.

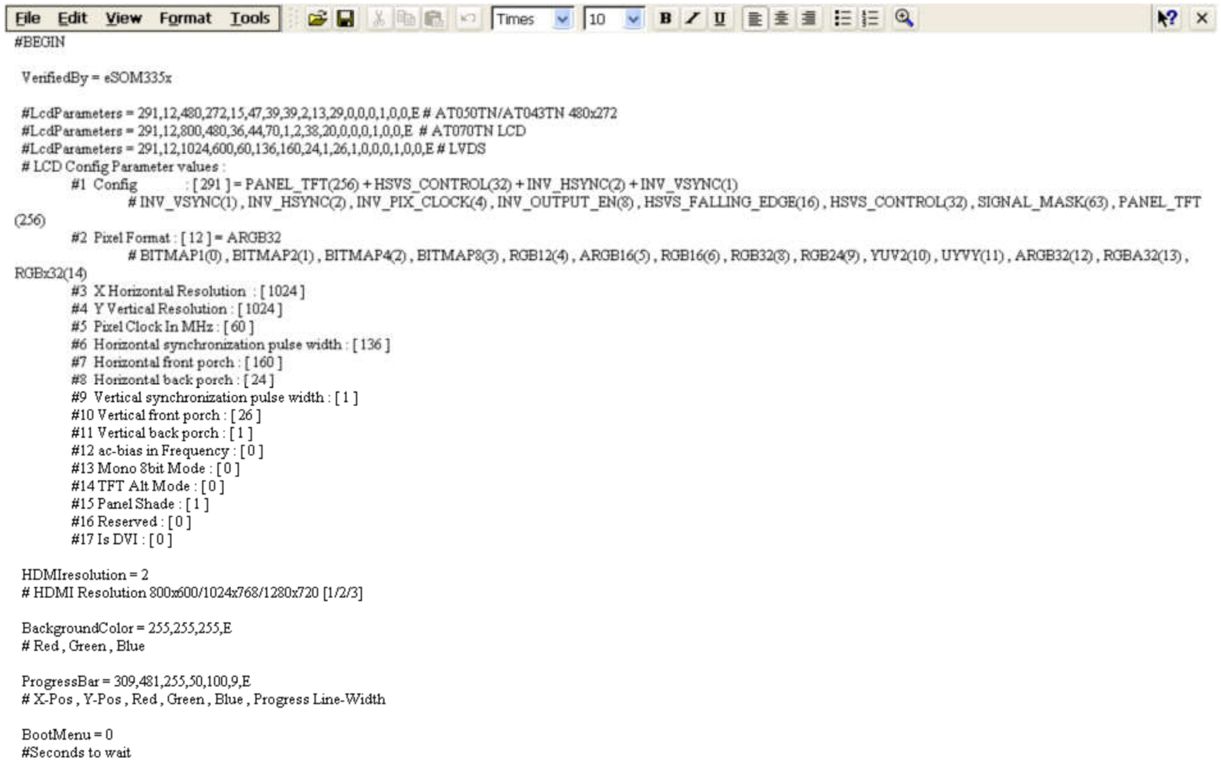
- **LOGO.BMP**: Users can show their logo when board boot WINCE OS. For this you should design your logo with BMP format with 32bit depth. In addition, your filename should be LOGO.BMP. In addition, you can design a frame for progressBar in LOGO.BMP file. An example is shown in below picture in Paint.net application:


LOGO.BMP - paint.net 4.2.14
File Edit View Image Layers Adjustments Effects
Tool: Brush width: 2 Hardness: 75% Fill: Solid Color Normal
Save Configuration
Settings
Preview, file size: 2.3 MB
Dithering level
Defaults
OK
Cancel
Colors
Primary
Left click to draw with primary color, right click to draw with secondary color.
1024 × 600
-295, -115

Bit Depth: 32-bit

File Name: LOGO.BMP
BMP format with 32bit depth

- **eSOM335x.ini:** Users can config display setting, boot background and ProgressBar via eSOM335x.ini file modification. Users can open this file in their device with WordPad application as a text file and change their configuration and save it. Another way for this work is modify eSOM335x.ini in PC and replace into this directory, anyway after restart, WinCE launched with new configuration. Available setting via eSOM335x.ini is described in the following.



```
#BEGIN

 VerifiedBy = eSOM335x

 #LcdParameters = 291,12,480,272,15,47,39,39,2,13,29,0,0,0,1,0,0,E # AT050TN/AT043TN 480x272
 #LcdParameters = 291,12,800,480,36,44,70,1,2,38,20,0,0,0,1,0,0,E  # AT070TN LCD
 #LcdParameters = 291,12,1024,600,60,136,160,24,1,26,1,0,0,0,1,0,0,E # LVDS
 # LCD Config Parameter values :
        #1  Config          : [ 291 ] = PANEL_TFT(256) + HSVS_CONTROL(32) + INV_HSYNC(2) + INV_VSYNC(1)
                # INV_VSYNC(1) , INV_HSYNC(2) , INV_PIX_CLOCK(4) , INV_OUTPUT_EN(8) , HSVS_FALLING_EDGE(16) , HSVS_CONTROL(32) , SIGNAL_MASK(63) , PANEL_TFT
(256)
        #2  Pixel Format : [ 12 ] = ARGB32
                # BITMAP1(0) , BITMAP2(1) , BITMAP4(2) , BITMAP8(3) , RGB12(4) , ARGB16(5) , RGB16(6) , RGB32(8) , RGB24(9) , YUV2(10) , UYVY(11) , ARGB32(12) , RGBA32(13) ,
RGBx32(14)
        #3  X Horizontal Resolution  : [ 1024 ]
        #4  Y Vertical Resolution : [ 1024 ]
        #5  Pixel Clock In MHz : [ 60 ]
        #6  Horizontal synchronization pulse width : [ 136 ]
        #7  Horizontal front porch : [ 160 ]
        #8  Horizontal back porch : [ 24 ]
        #9  Vertical synchronization pulse width : [ 1 ]
        #10 Vertical front porch : [ 26 ]
        #11 Vertical back porch : [ 1 ]
        #12 ac-bias in Frequency : [ 0 ]
        #13 Mono 8bit Mode : [ 0 ]
        #14 TFT Alt Mode : [ 0 ]
        #15 Panel Shade : [ 1 ]
        #16 Reserved : [ 0 ]
        #17 Is DVI : [ 0 ]

 HDMIresolution = 2
 # HDMI Resolution 800x600/1024x768/1280x720 [1/2/3]

 BackgroundColor = 255,255,255,E
 # Red , Green , Blue

 ProgressBar = 309,481,255,50,100,9,E
 # X-Pos , Y-Pos , Red , Green , Blue , Progress Line-Width

 BootMenu = 0
 #Seconds to wait
```

Note: Command in this script file contain following segments:
- **Parameter Name:** each command begins with parameter name
- **Parameter Values:** argument of each values
- **End of Command:** **‘**E’ character shows the end of command.

<u>Display Setting</u>: Users can change LCD display easily without challenging with OS customization. It could be done through ***eSOM335x.ini*** file. Open this file and configure your LCD with LcdParameters values.

If LCD has not included in your project, comment LcdParameters line with # character. in this case device bootloader consider HDMI as Display port.

For determine your HDMI resolution, you must change ***HDMIresolution*** , by default this parameter is settled to 2

number	Resolution
1	800x600
2	1024x768
3	1280x720

If you want using LCD in your project, you should edit LcdParameters line. An example for TFT LCD:

LcdParameters = 291,12,1024,600,60,136,160,24,1,26,1,0,0,0,1,0,0,E

Parameters for 4 type of LCDs is written for users in this file previously, and if users select any of them, they should uncomment them by remove # sign and comment others. LCD 4.3” and 5” have same config parameters. Following table shows configurations parameters in LcdParameters command.

Number	Description	Values

1	Config	Config: [291] = PANEL_TFT(256) + HSVS_CONTROL(32) + INV_HSYNC(2) + INV_VSYNC(1)
		INV_VSYNC(1)
		INV_HSYNC(2)
		INV_PIX_CLOCK(4)
		INV_OUTPUT_EN(8)
		HSVS_FALLING_EDGE(16)
		HSVS_CONTROL(32)
		SIGNAL_MASK(63)
		PANEL_TFT(256)

2	Pixel Format	This parameter value is between 0 to 14. Value can be selected from below list according to LCD Type:
		BITMAP1(0)
		BITMAP2(1)
		BITMAP4(2)
		BITMAP8(3)
		RGB12(4)
		ARGB16(5)
		RGB16(6)
		RGB32(8)
		RGB24(9)
		YUV2(10)
		UYVY(11)
		ARGB32(12)
		RGBA32(13)
		RGBx32(14)

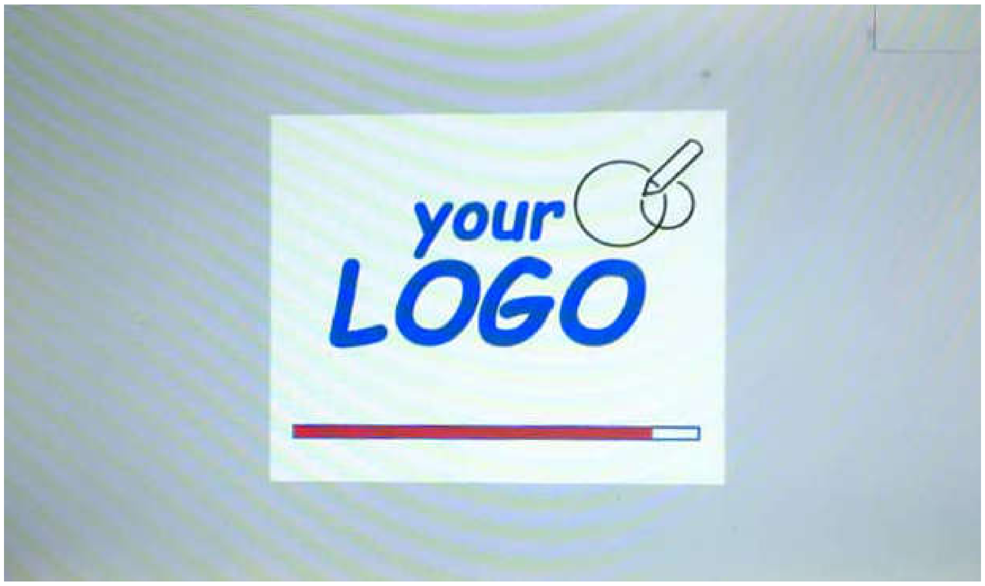
3	X Horizontal Resolution	From LCD datasheet
4	Y Vertical Resolution	From LCD datasheet
5	Pixel Clock In MHz	From LCD datasheet
6	Horizontal synchronization pulse width	From LCD datasheet
7	Horizontal front porch	From LCD datasheet
8	Horizontal back porch	From LCD datasheet
9	Vertical synchronization pulse width	From LCD datasheet
10	Vertical front porch	From LCD datasheet
11	Vertical back porch	From LCD datasheet
12	ac-bias in Frequency	From LCD datasheet
13	Mono 8bit Mode	
14	TFT Alt Mode	
15	Panel Shade	
16	Reserved	
17	Is DVI	

- ***Background Color:*** User can determine background color when boot OS with ***BackgroundColor*** parameter. For example:

$$\text{BackgroundColor} = \underbrace{200}_{\text{Red Value}},\underbrace{210}_{\text{Green Value}},\underbrace{220}_{\text{Blue Value}},\underbrace{E}_{\text{End}}$$

With this setting, background color will change to ▬ (Red=200 Green=210, Blue=220) color as shown as below picture:

A good reference for RGB values is www.w3schools.com/colors/colors_rgb.asp website or using a graphic softwares like Paint.net.

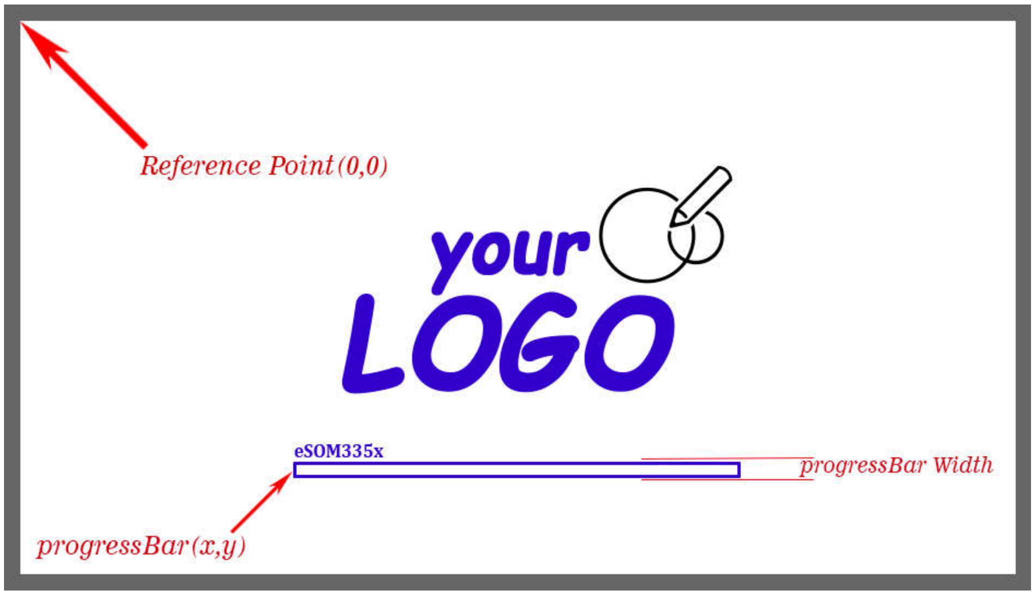


- ***Progress Bar Setting*** : In addition of boot logo, user can customize boot progress bar and own their personal boot image and progress bar. This can be implemented by changing ***ProgressBar*** parameter in the ***eSOM335x.ini*** file. Table below shows parameters should be changed. Point (0,0) is top left corner of LCD.

$$\text{ProgressBar} = \underbrace{327}_{\text{x Position}},\underbrace{482}_{\text{y position}},\underbrace{255}_{\text{Red Value}},\underbrace{50}_{\text{Green Value}},\underbrace{100}_{\text{Blue Value}},\underbrace{10}_{\text{Thickness}},\underbrace{E}_{\text{End}}$$

Number	Parameter	Description
1	X-Pos	Determine where progress Bar start in x direction.
2	Y-Pos	Determine where progress Bar start in y direction.
3	Red	Determine the amount of red in progress bar color.
4	Green	Determine the amount of green in progress bar color.
5	Blue	Determine the amount of blue in progress bar color.
6	Progress Linewidth	Determine the progress bar line width.

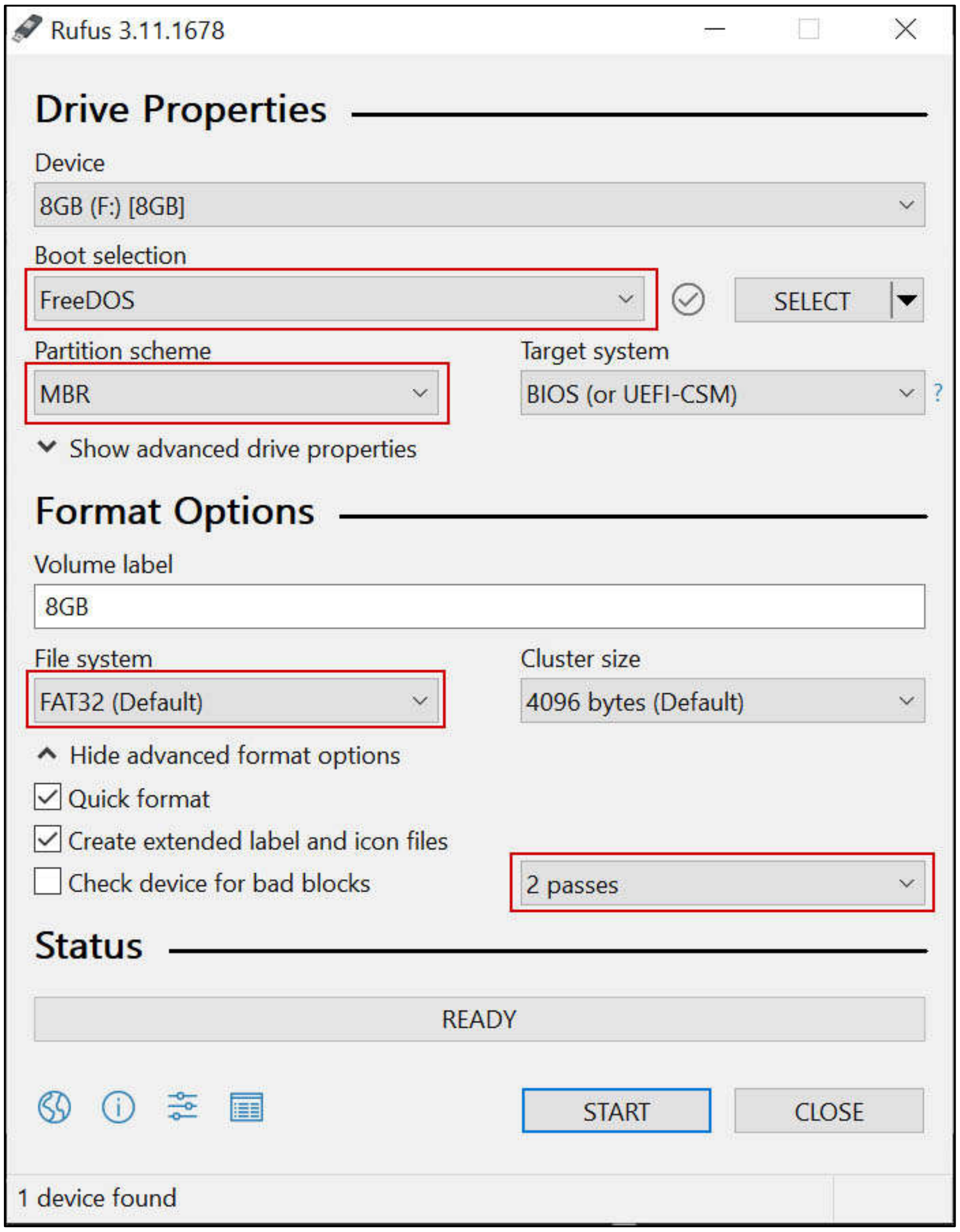
Following figure shows results for these settings. You can design a frame for progressBar in LOGO.BMP file.



- **NK.BIN:** users can customize WinCE kernel via NK.BIN file, through Platform Builder building routine described in this document, after building your NK.BIN, you can replace existing NK.BIN with your NK.BIN, after restart, WinCE launch with new kernel.

Prepare microSD

To prepare your microSD card, you'll need a computer with ability to read and write SD cards. For boot from microSD, firstly microSD is formatted with FAT32 by click on uSD drive and click on format. Another way is using ***Rufus*** application. Setting for this application is shown as following picture, click on ***START*** to format your uSD. After format microSD, delete all files on uSD if exist.



Rufus 3.11.1678

Drive Properties

Device: 8GB (F:) [8GB]

Boot selection: FreeDOS

Partition scheme: MBR

Target system: BIOS (or UEFI-CSM)

Show advanced drive properties

Format Options

Volume label: 8GB

File system: FAT32 (Default)

Cluster size: 4096 bytes (Default)

Hide advanced format options

[✓] Quick format

[✓] Create extended label and icon files

[ ] Check device for bad blocks — 2 passes

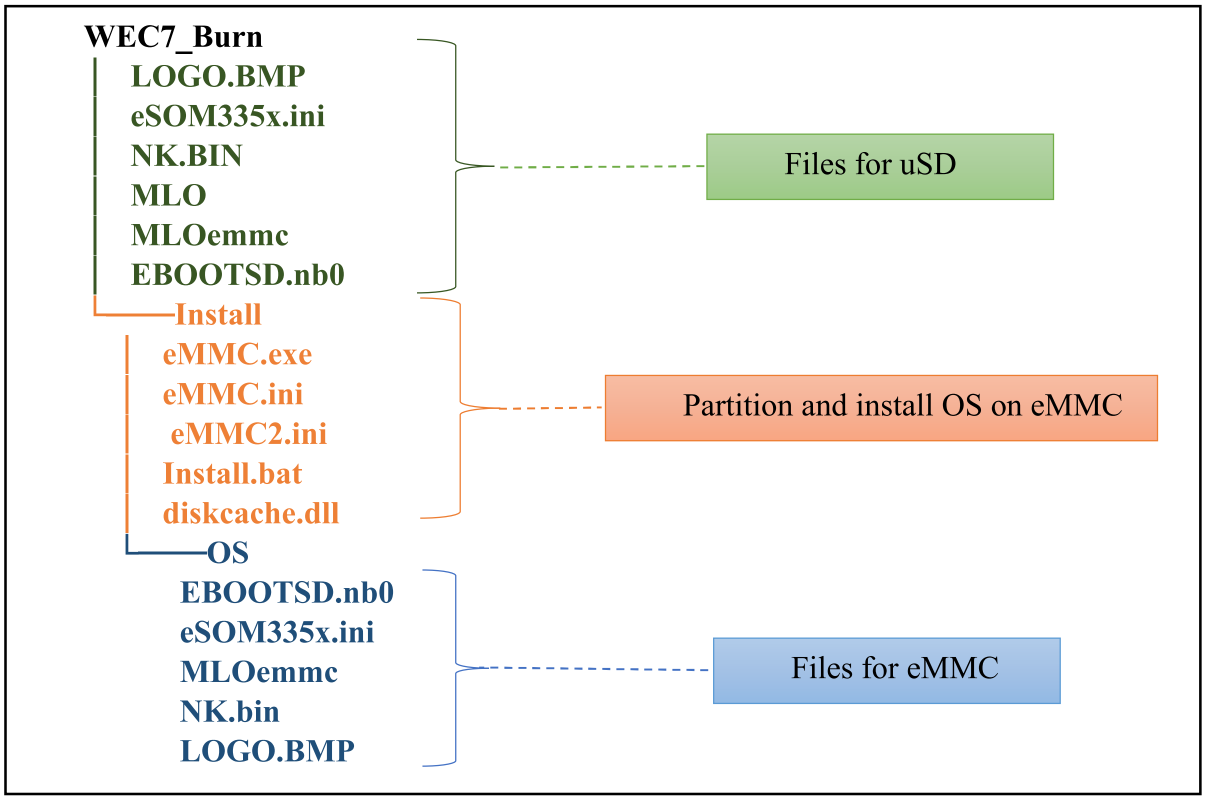
Status

READY

START CLOSE

1 device found

In ***WEC7_Burn*** folder of your net disk URL exist files for eMMC booting and microSD as shown in below image, these files have the same name in uSD root directory but with the different configuration for OS booting. uSD files is based on a RAM based OS, but eMMC files is based on Hive base OS.



Note: These files are for boot from uSD (green colored text) and eMMC (blue colored text). If users just want to create a bootable microSD green colored file is sufficient, they should copy these files into their microSD card. But if users want to burn both eMMC module and microSD, they should copy all of these files into microSD card, then follow following instructions:

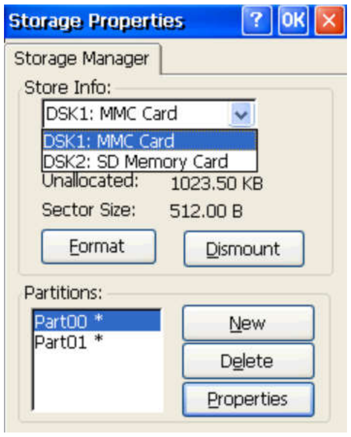
Note: User can config display and logo setting via ***eSOM335x.ini*** files described earlier in Boot Customization section, before copying these files, there are two ***eSOM335x.ini*** file one for uSD and one for eMMC, therefore setting for them might be different according to user requirements.

- Copy corresponding files to your microSD card root folder
- Insert your microSD into your development board and power on it.
- If you use eDVK335x Press Boot Key then press RST Key on this development board to CPU boot from uSD.
- After a while WINCE Desktop appear.
- Enjoy it.

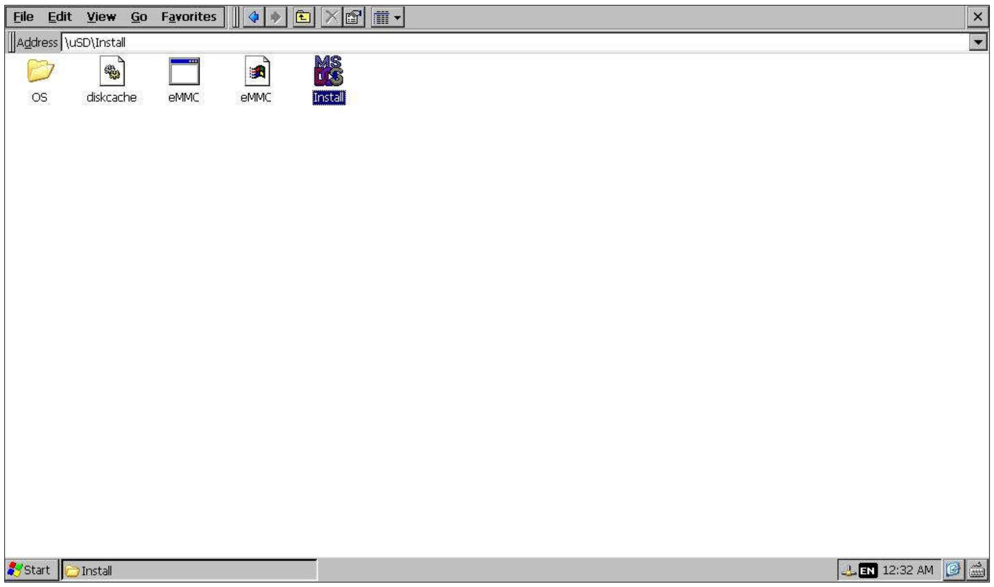
Prepare eMMC

The eSOM335x comes with preinstalled WinCE in its eMMC module, but if user want to prepare a new eMMC module, follow this instruction:

- **Via MicroSD:** In this way, you need to copy install folder in your microSD card root and insert into your development board uSD slot. After eSOM335x boots with your microSD card, firstly check which dive is associated with your eMMC drive, therefore go to ***Control Panel*** then click on ***Storage Manager*** if Disk1 is associated with SD card rename ***eMMC*** file on Install folder to ***eMMC1*** and then rename ***eMMC2*** to ***eMMC***.



- go to your microSD drive and open install folder and run install file.



In this case, OS partitioning eMMC and copy necessary files in ***uSD/install/OS*** folder to eMMC drive. If reset your development board, it boots from eMMC.

Note: if you want to upgrade your OS in eMMC, you can just copy your ***NK.bin*** file to eMMC and replace with old version.

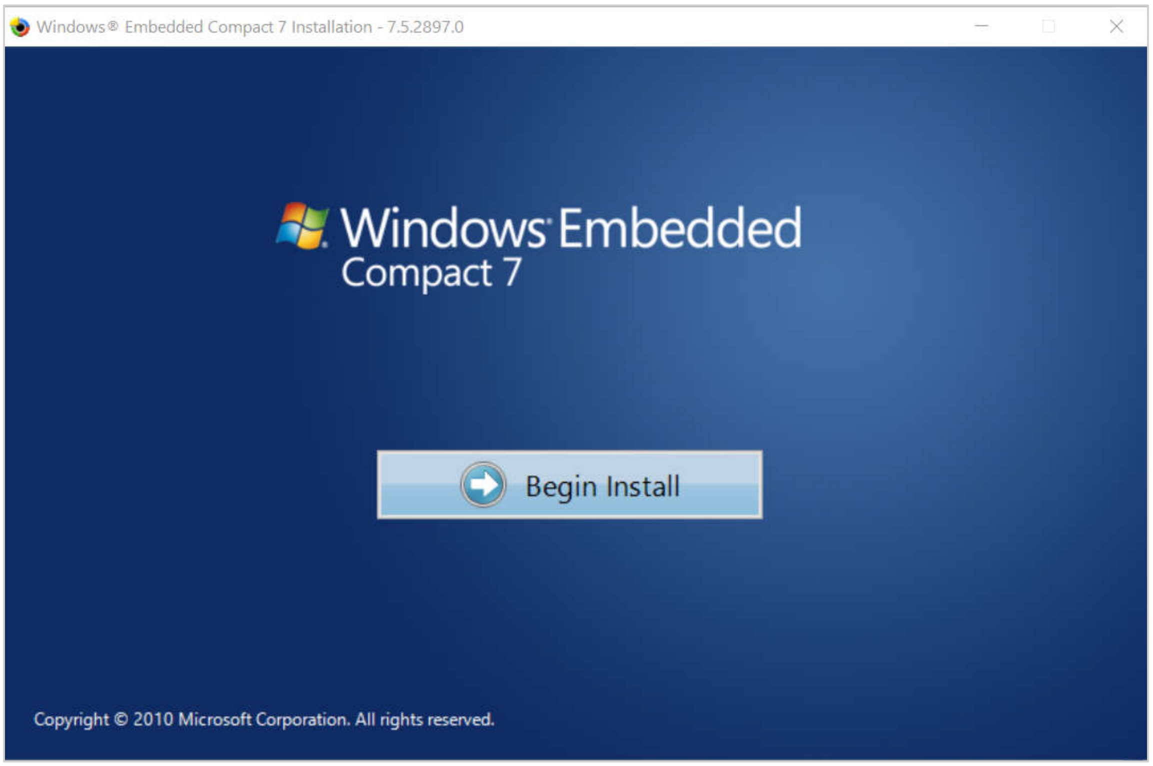
Platform Builder

Platform Builder for Windows Embedded Compact 7 and Visual Studio 2008 provide a graphical user interface for creating Compact 7 OS designs. Platform Builder, included with the Windows Embedded tools, customizes the Visual Studio integrated development environment (IDE) and you can build custom embedded OS designs based on the Windows Embedded CE OS. Platform Builder for Windows Embedded Compact 7 and Visual Studio 2008 provide a graphical user interface for creating Compact 7 OS designs. Together, Platform Builder and Visual Studio make up a complete integrated development environment (IDE) where you can build, develop, download, and debug code for an embedded device all from a single environment. So, it is possible to customize the Windows Embedded Compact Image and write user applications with just one tool. Platform Builder includes an OS design wizard that makes it easy to select the components that you'll need for your OS. These components include redefined Board Support Packages (BSPs) and design templates containing catalog items for the functionality that your OS can support.

Installation

For install Platform builder follow these instructions:

- Firstly, install Visual studio 2008.
- Open ***Windows Embedded Compact 7*** folder and install it.
- If in installation process, needs to path of any file, select ***Windows Embedded Compact 7*** folder on your PC.

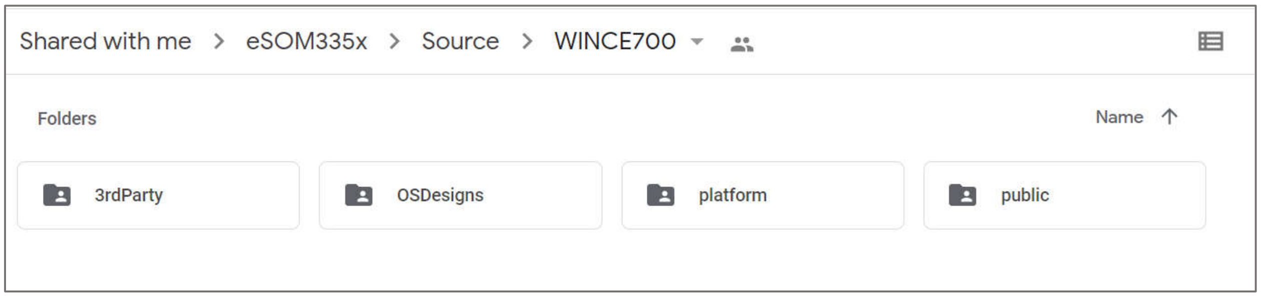

Windows® Embedded Compact 7 Installation - 7.5.2897.0
Windows Embedded
Compact 7
Begin Install
Copyright © 2010 Microsoft Corporation. All rights reserved.

NOTE: If Platform Builder expired after a time, you can change your PC date to a date before your Platform Builder.

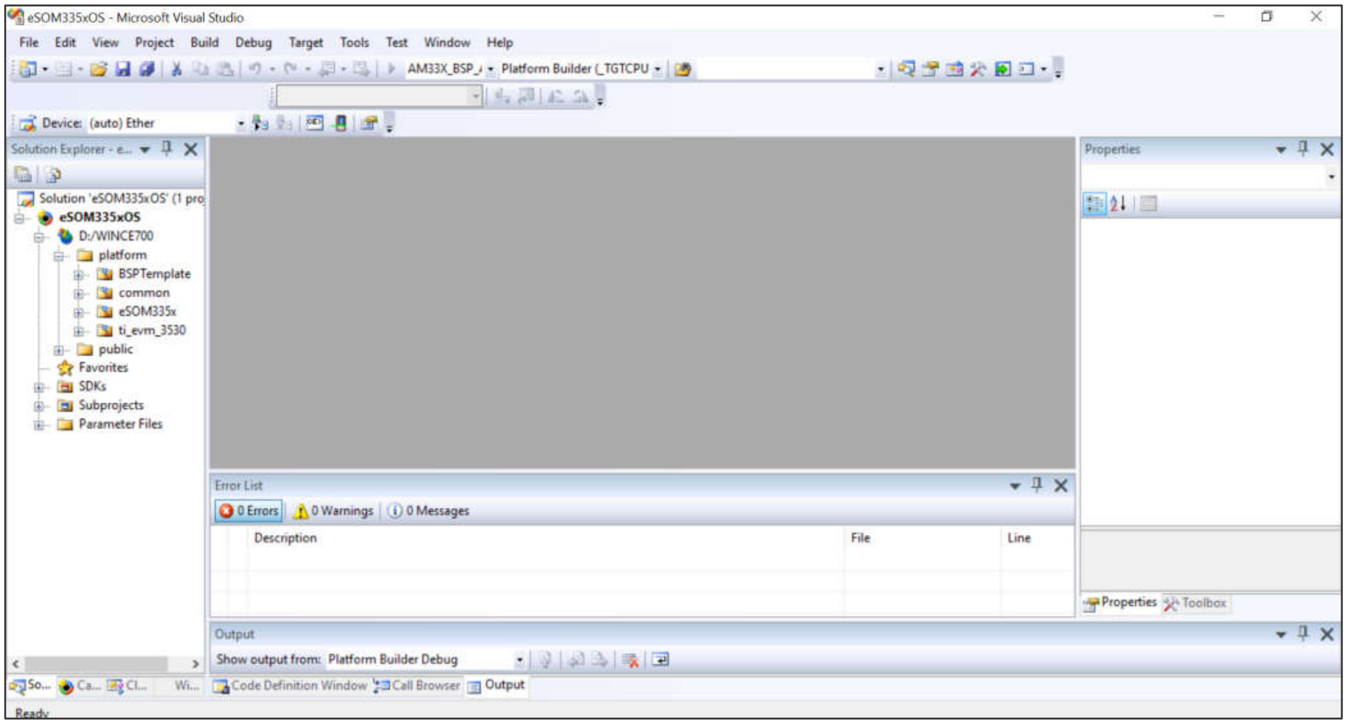
Configure OS

The process of designing an OS includes creating the initial OS design, and then modifying and refining the design until the OS has all the functionality and support required for the target device. A design template is a possible starting point for an OS design. You can begin an OS design by choosing a design template, or you can begin an OS design by selecting catalog items individually, depending on the requirements for your device. The eSOM335x come with a configured project and source code for work with Platform Builder.

- After install Platform Builder, copy ***WINCE700*** content files (from Source folder in net disk drive URL) to WINCE700 folder that created when Platform Builder installation.



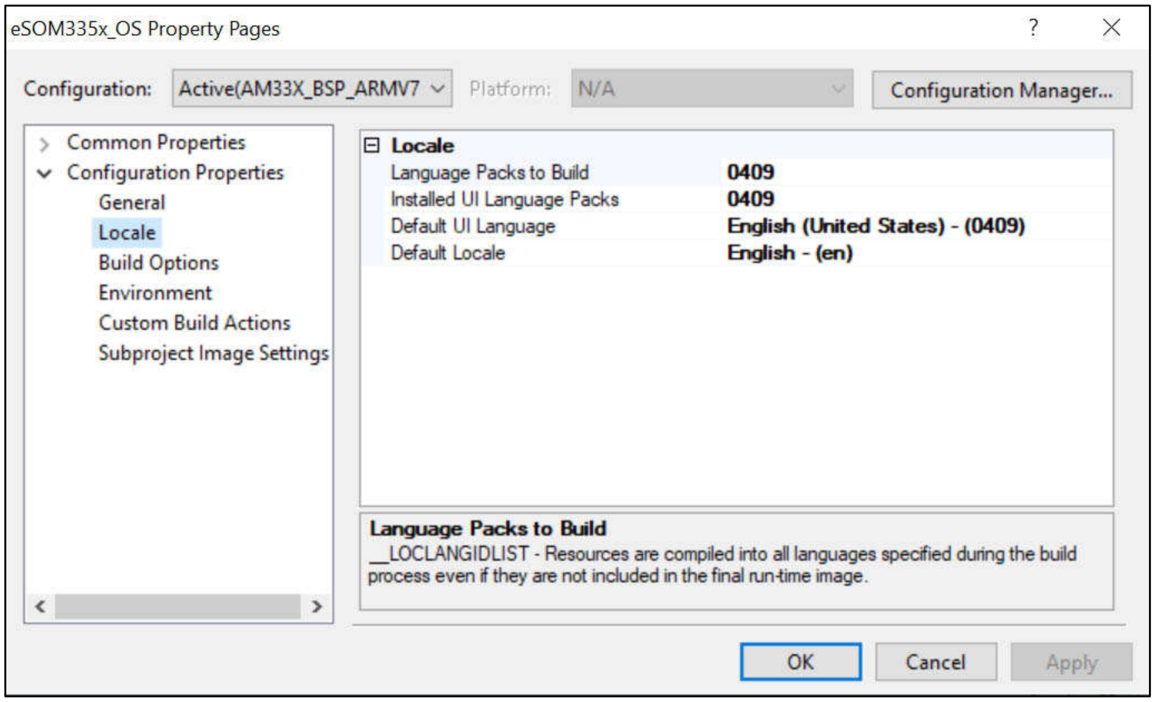
- To build using the Platform Builder user interface (UI), you must create an OS design or open an existing one. A complete project is created and exist in ***WINCE700*** folder. In this folder navigate to ***\OSDesigns\eSOM335xOS*** and click on ***eSOM335xOS.sln*** for open it.



- The Windows Embedded Compact 7 build system is responsible for building an entire operating system. Unlike building a software application, building an entire operating system includes a large variety of components:
  - The core operating system (kernel, file system, security features)
  - Built-in applications and services (Media Player, Internet Explorer)
  - Built-in driver stacks (USB, networking)
  - Custom drivers and hardware abstractions (display driver, audio driver)
  - Custom applications

The first objective of the build system is to combine all of these components into a single binary image. An image is typically a single file that contains all of the above components laid out in memory so that they can be downloaded to a device and then executed by the device. The second objective of the build system is to allow components to be easily selected and substituted so that only the necessary components are included in the image.

- Platform Builder gives you several options for building your OS design. You can use the Locale page to set environment variables that are passed to the build localization tools during the build process. In this way, you can create an OS design for non-English languages. For add a UI language for Windows environment right click on eSOM335xOS, then click on ***Properties***, in opened window go to Locale Tab and click on ***Language Packs to Build*** and tick your language checkbox.



Locale

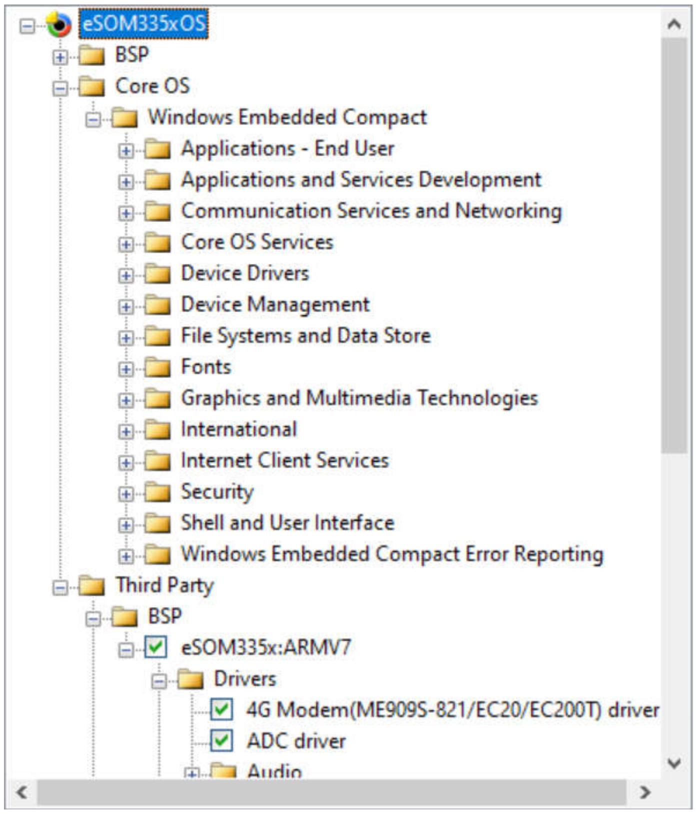
Language Packs to Build	0409
Installed UI Language Packs	0409
Default UI Language	English (United States) - (0409)
Default Locale	English - (en)

Language Packs to Build
__LOCLANGIDLIST - Resources are compiled into all languages specified during the build process even if they are not included in the final run-time image.

- In ***Catalog Item View*** you can do additional configuration in this section. In fact, catalog is a graphical representation of all the available components in the Compact 7 OS. Typically, these components are not hardware specific. Each OS design has a set of catalog items associated with it. By selecting catalog items, you can include or exclude OS functionality. When you exclude components, they are not built by the build process or included in the final image. ***Catalog Items View*** consists of 3 section:

- BSP
- Core OS
- Third Party

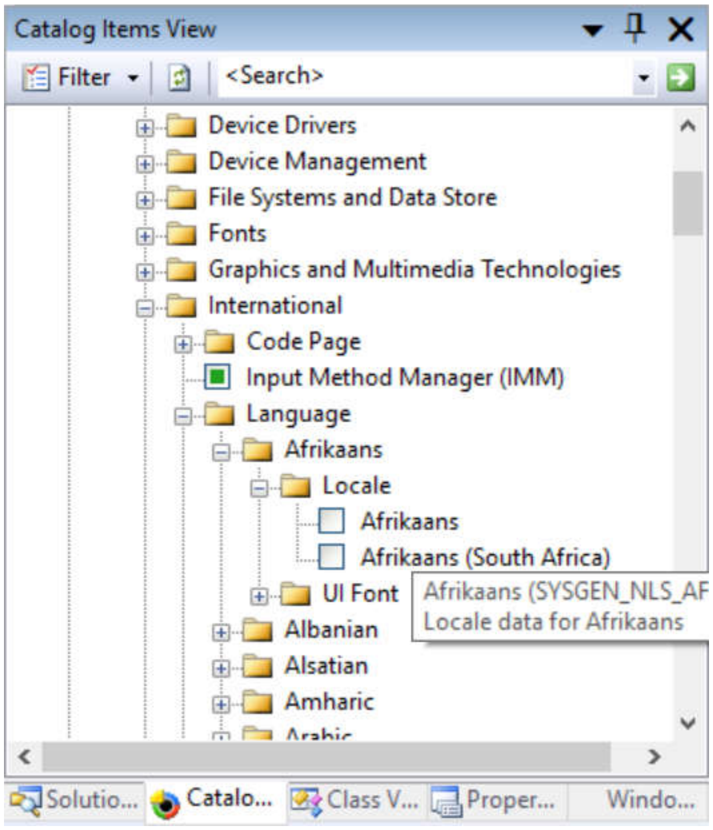
Your initial catalog item selections are simply a starting point for your OS design. You can add and remove catalog items throughout development.



- With click on Core OS, the hierarchy list of all available catalog items is shown that you can configured for your customized OS. Device Driver , .Net Compact Framework, Fonts, … are components can be modified depending on the user needs. Select a catalog item to add it or clear the check box to remove the catalog item. Some of these items dependent on another items, be care for deselect these options. Selected catalog items are explicitly included. A green box next to a catalog item(🟩) indicates that another catalog item depends on it; the catalog item with the green box will be included as long as its dependent item is included.

Add Language

for adding another language to your OS through ***Platform Builder***, in catalog item tab in ***Core OS*** section click on ***International***, then click ***Language***, list of world language opened, and user can tick them upon their project needs. Some language needs to ticked ***Unicode Script Processor for Complex Scripts*** check box to support Unicode's in your device.

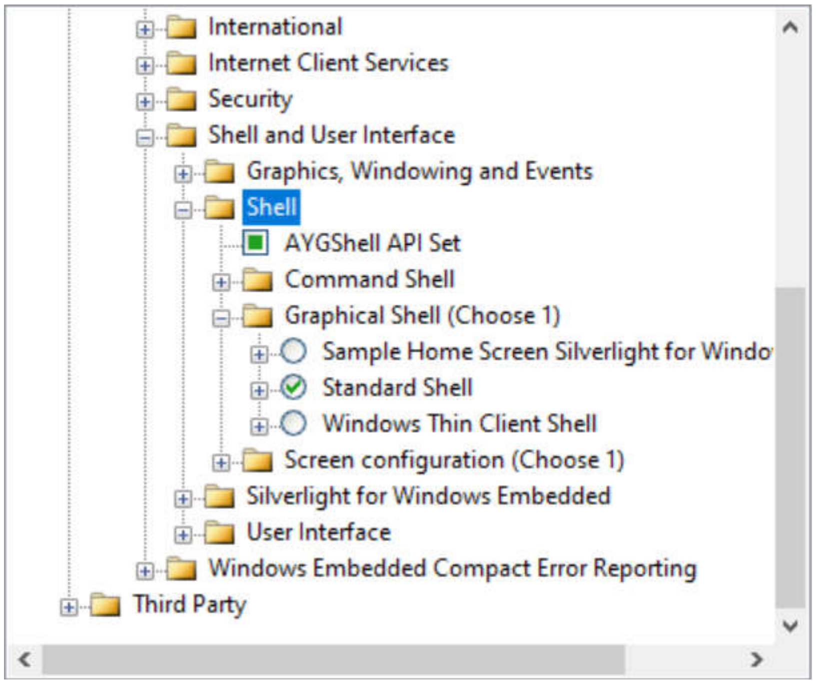


Afrikaans (SYSGEN_NLS_AF
Locale data for Afrikaans

Shell and User Interface

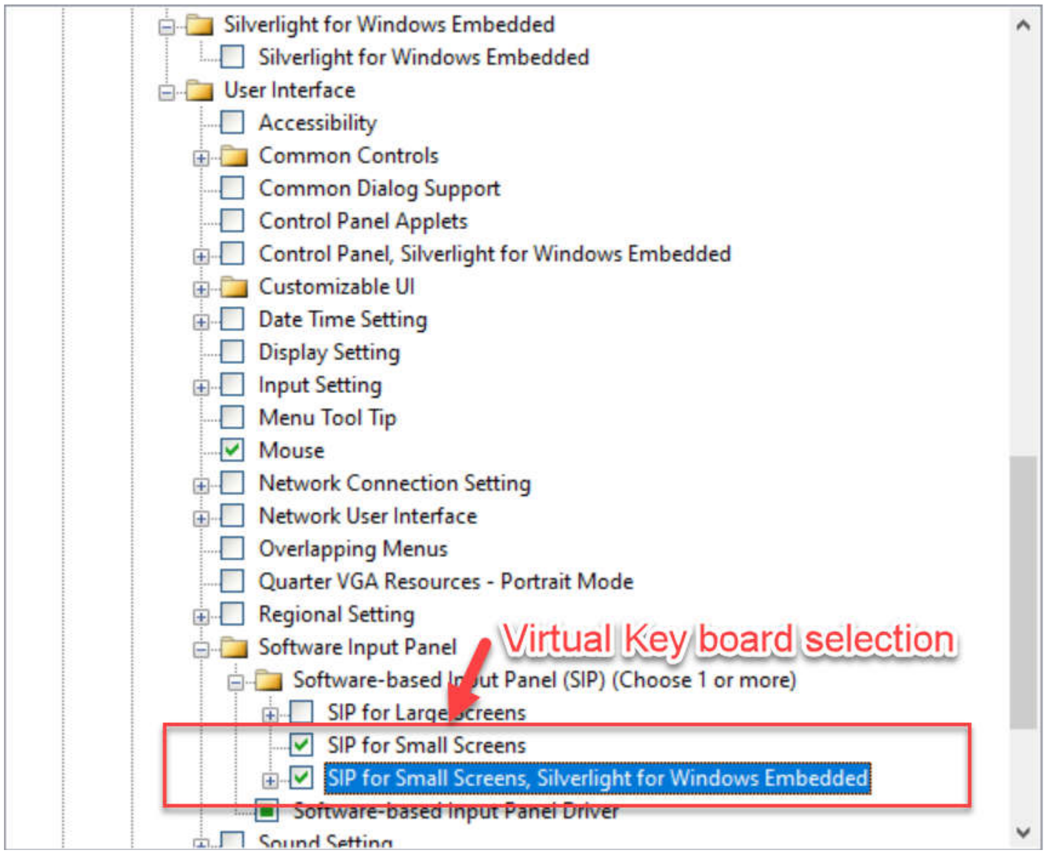
in this section you can select your Windows UI. ***Standard Shell*** is selected for Windows UI by default. You can deselect this, if you do not need to any Shell, then embed your application for running it after OS booting.

Note: ***Smart Devices*** Library depend on ***Standard Shell.*** For remove Shell, you should remove catalog item "Smart Devices" as well as "Standard Shell".



- International
- Internet Client Services
- Security
- Shell and User Interface
  - Graphics, Windowing and Events
  - Shell
    - AYGShell API Set
    - Command Shell
    - Graphical Shell (Choose 1)
      - Sample Home Screen Silverlight for Windo
      - ✓ Standard Shell
      - Windows Thin Client Shell
    - Screen configuration (Choose 1)
  - Silverlight for Windows Embedded
  - User Interface
- Windows Embedded Compact Error Reporting
- Third Party

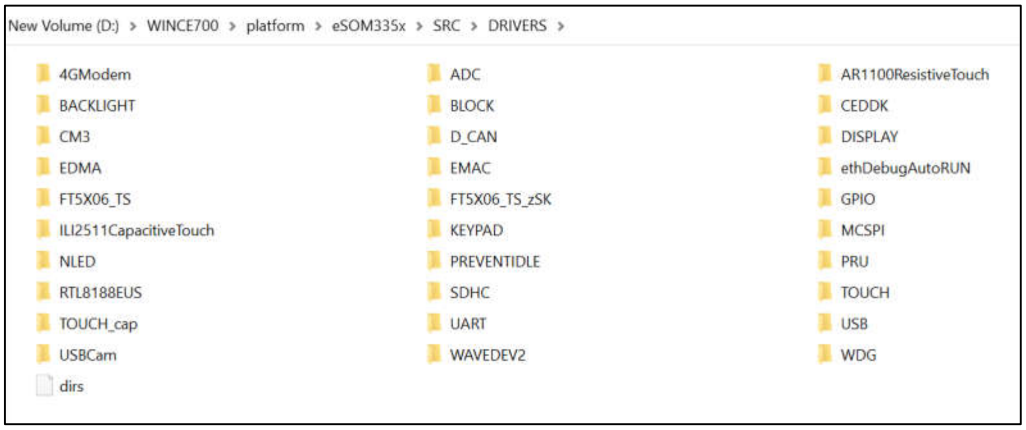
By default, two types of virtual keyboards is ready for user interface, users can customize virtual keyboard through select/deselect them in ***User Interface*** as shown on below figure:



- Silverlight for Windows Embedded
  - Silverlight for Windows Embedded
- User Interface
  - Accessibility
  - Common Controls
  - Common Dialog Support
  - Control Panel Applets
  - Control Panel, Silverlight for Windows Embedded
  - Customizable UI
  - Date Time Setting
  - Display Setting
  - Input Setting
  - Menu Tool Tip
  - ✓ Mouse
  - Network Connection Setting
  - Network User Interface
  - Overlapping Menus
  - Quarter VGA Resources - Portrait Mode
  - Regional Setting
  - Software Input Panel
    - Software-based Input Panel (SIP) (Choose 1 or more)
      - SIP for Large Screens
      - ✓ SIP for Small Screens
      - ✓ SIP for Small Screens, Silverlight for Windows Embedded
    - Software-based Input Panel Driver
  - Sound Setting

BSP

Many drivers are developed for the eSOM335x and users can include or exclude them into their projects. List of drivers are included in BSP source as shown in following figure:

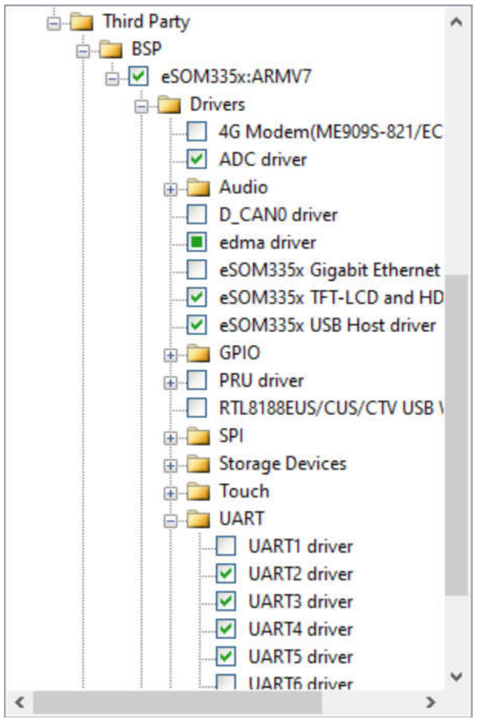


New Volume (D:) > WINCE700 > platform > eSOM335x > SRC > DRIVERS >

4GModem	ADC	AR1100ResistiveTouch
BACKLIGHT	BLOCK	CEDDK
CM3	D_CAN	DISPLAY
EDMA	EMAC	ethDebugAutoRUN
FT5X06_TS	FT5X06_TS_zSK	GPIO
ILI2511CapacitiveTouch	KEYPAD	MCSPI
NLED	PREVENTIDLE	PRU
RTL8188EUS	SDHC	TOUCH
TOUCH_cap	UART	USB
USBCam	WAVEDEV2	WDG
dirs		

BSP catalog in ***Third Party*** section containing hardware-specific drivers and other routines that allow a particular operating system to function in a particular hardware environment. Users can select/deselect any hardware driver according to **pinMux** document and their project requirements.

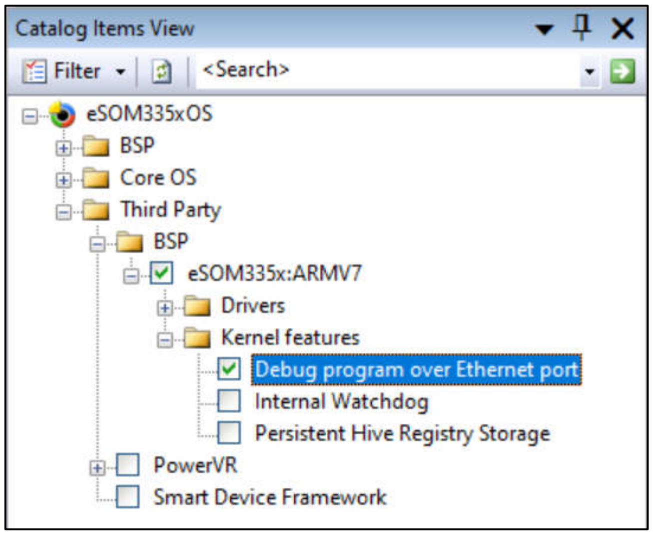
For example, users can enable UART1. The eSOM335x has 5 UART, by default UART1 is reserved for debug, but users can enable this port for other application with tick UART1 check box.



Third Party
BSP
☑ eSOM335x:ARMV7
Drivers
☐ 4G Modem(ME909S-821/EC
☑ ADC driver
Audio
☐ D_CAN0 driver
■ edma driver
☐ eSOM335x Gigabit Ethernet
☑ eSOM335x TFT-LCD and HD
☑ eSOM335x USB Host driver
GPIO
☐ PRU driver
☐ RTL8188EUS/CUS/CTV USB \
SPI
Storage Devices
Touch
UART
☐ UART1 driver
☑ UART2 driver
☑ UART3 driver
☑ UART4 driver
☑ UART5 driver
☐ UART6 driver

The pinMux document is associated to pads config file (***bsp_padcfg.h***) existed ***Source\WINCE700\platform\eSOM335x\SRC\INC*** folder.

Another option in BSP directory is ***Debug Program over Ethernet*** option that it is if selected, users can deploy and debug application from Host PC to their boards.

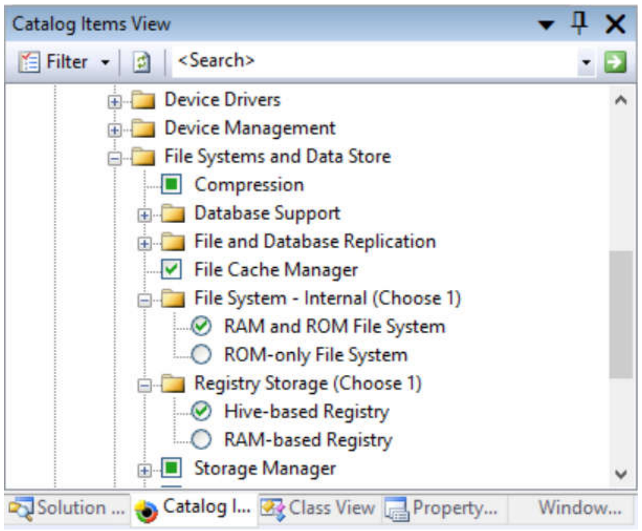


- eSOM335xOS
  - BSP
  - Core OS
  - Third Party
    - BSP
      - [✓] eSOM335x:ARMV7
        - Drivers
        - Kernel features
          - [✓] Debug program over Ethernet port
          - [ ] Internal Watchdog
          - [ ] Persistent Hive Registry Storage
    - [ ] PowerVR
    - [ ] Smart Device Framework

File Systems and Data Store

The WINCE registry holds the data of applications, drivers, user parameter configurations and other configuration settings. There is two possible ways to provide for registry persistence in a Windows CE system:

- ***Hive Based Registry:*** Save the changes to the registry in files on a disk.
- ***RAM Based Registry:*** Saves the changes to the registry in RAM.



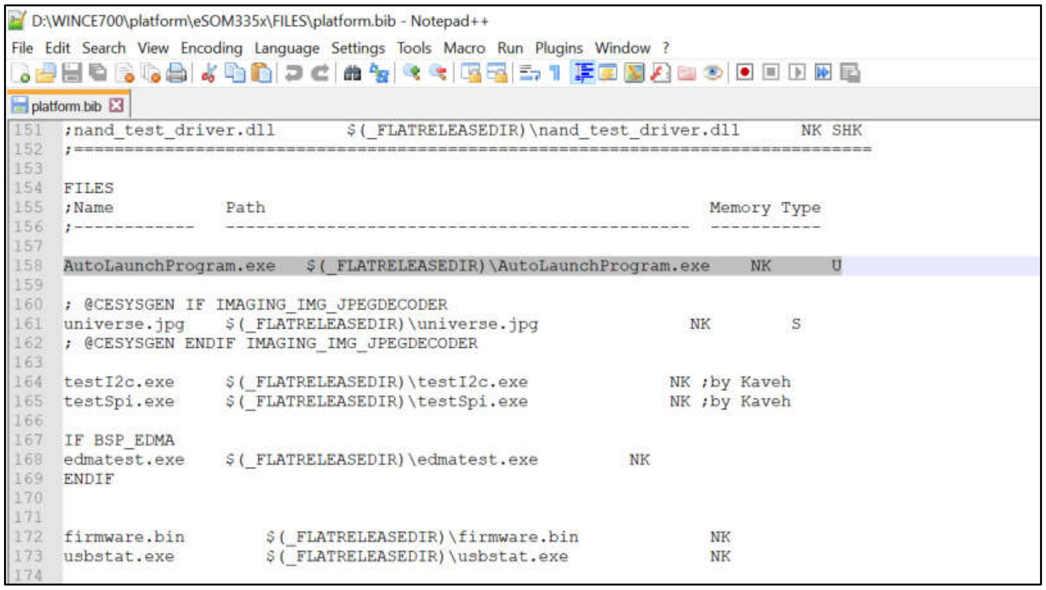
- Device Drivers
- Device Management
- File Systems and Data Store
  - Compression
  - Database Support
  - File and Database Replication
  - [✓] File Cache Manager
  - File System - Internal (Choose 1)
    - [✓] RAM and ROM File System
    - [ ] ROM-only File System
  - Registry Storage (Choose 1)
    - [✓] Hive-based Registry
    - [ ] RAM-based Registry
  - Storage Manager

In RAM-based method if the power supply of the Board is powered off the registry data based on RAM will be lost, In Hive-based method, all changes saved in system and that is not lost when the power is turned off. The registry can also be used for system initialization like auto launch an Application.

Adding file or application to OS Structure

If you want to embed an application in your BSP, follow this instruction:

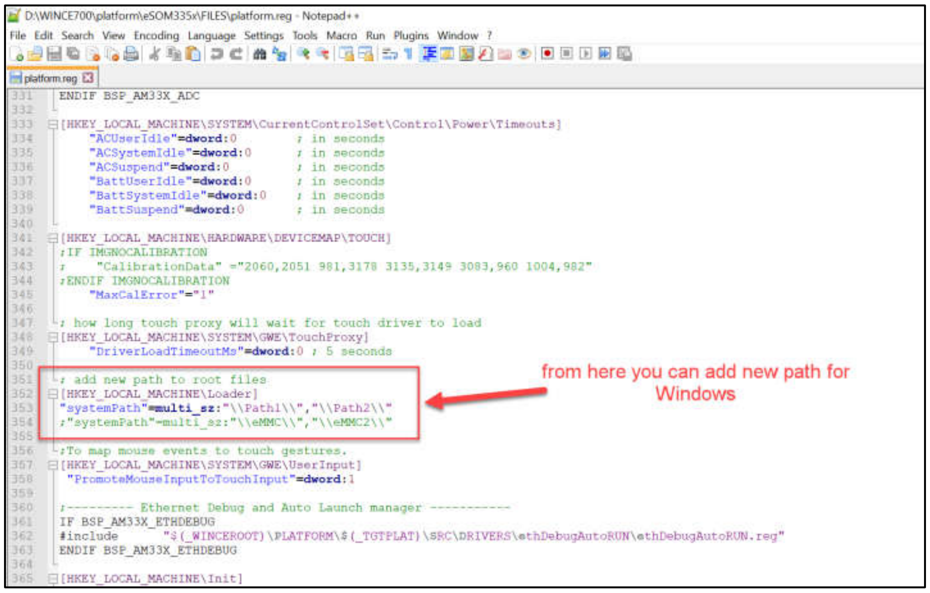
- Open ***WINCE700\platform\eSOM335x\FILES*** folder and paste your file here.
- Open ***platform.bib*** file and write your program name in this file, in example ***AutoLaunchProgram.exe*** is added to this file in line 158 as shown in following figure:



D:\WINCE700\platform\eSOM335x\FILES\platform.bib - Notepad++

```
151 ;nand_test_driver.dll       $(_FLATRELEASEDIR)\nand_test_driver.dll      NK SHK
152 ;===============================================================================
153
154 FILES
155 ;Name            Path                                              Memory Type
156 ;------------    ----------------------------------------------    -----------
157
158 AutoLaunchProgram.exe    $(_FLATRELEASEDIR)\AutoLaunchProgram.exe     NK       U
159
160 ; @CESYSGEN IF IMAGING_IMG_JPEGDECODER
161 universe.jpg     $(_FLATRELEASEDIR)\universe.jpg              NK        S
162 ; @CESYSGEN ENDIF IMAGING_IMG_JPEGDECODER
163
164 testI2c.exe      $(_FLATRELEASEDIR)\testI2c.exe             NK ;by Kaveh
165 testSpi.exe      $(_FLATRELEASEDIR)\testSpi.exe             NK ;by Kaveh
166
167 IF BSP_EDMA
168 edmatest.exe     $(_FLATRELEASEDIR)\edmatest.exe          NK
169 ENDIF
170
171
172 firmware.bin         $(_FLATRELEASEDIR)\firmware.bin            NK
173 usbstat.exe          $(_FLATRELEASEDIR)\usbstat.exe             NK
174
```

- By default, ***Windows*** folder is your board path, therefore most of applications exist in this folder, that means, after calling, ***Windows*** can find them. In addition, users can add another path to OS, if users want to add folders to in their BSPs via Platform Builder, they can open **platform.reg** file in ***WINCE700\platform\eSOM335x\FILES*** folder and write their paths instead of "***Path1***" and "***Path2***" as shown in following picture, then in platform builder after all modifications, build new OS. In this way, in new NK image, new path is added to Windows and users can check this in ***Loader*** section of ***Registry Editor***.

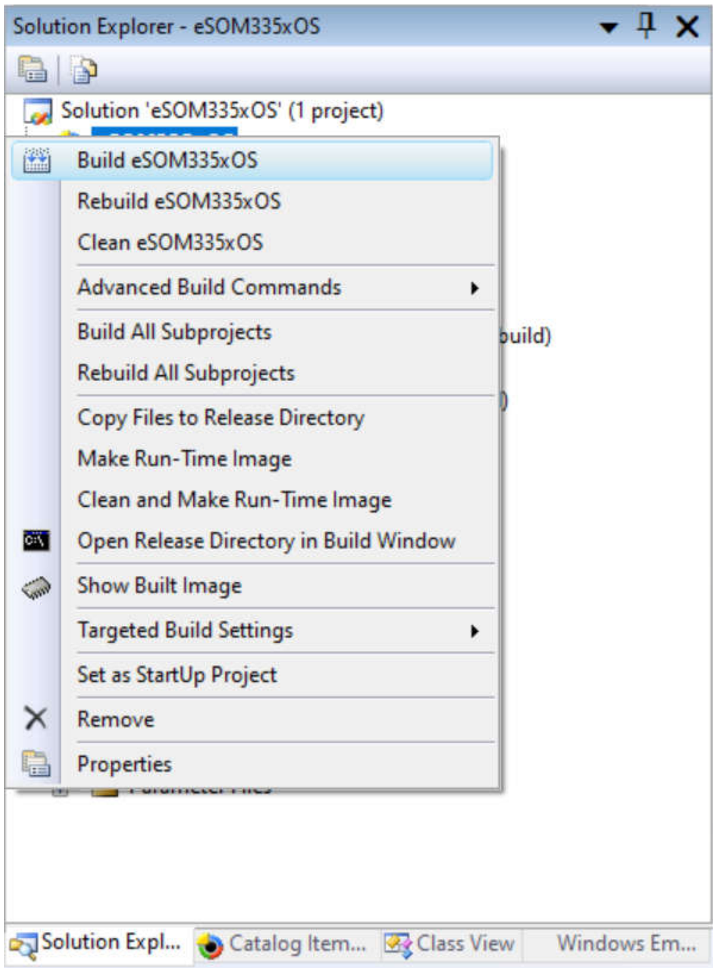


D:\WINCE700\platform\eSOM335x\FILES\platform.reg - Notepad++

```
ENDIF BSP_AM33X_ADC

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Power\Timeouts]
    "ACUserIdle"=dword:0          ; in seconds
    "ACSystemIdle"=dword:0        ; in seconds
    "ACSuspend"=dword:0           ; in seconds
    "BattUserIdle"=dword:0        ; in seconds
    "BattSystemIdle"=dword:0      ; in seconds
    "BattSuspend"=dword:0         ; in seconds

[HKEY_LOCAL_MACHINE\HARDWARE\DEVICEMAP\TOUCH]
;IF IMGNOCALIBRATION
;     "CalibrationData" ="2060,2051 981,3178 3135,3149 3083,960 1004,982"
;ENDIF IMGNOCALIBRATION
    "MaxCalError"="1"

; how long touch proxy will wait for touch driver to load
[HKEY_LOCAL_MACHINE\SYSTEM\GWE\TouchProxy]
    "DriverLoadTimeoutMs"=dword:0 ; 5 seconds

; add new path to root files
[HKEY_LOCAL_MACHINE\Loader]
"systemPath"=multi_sz:"\\Path1\\","\\Path2\\"
;"systemPath"=multi_sz:"\\eMMC\\","\\eMMC2\\"

;To map mouse events to touch gestures.
[HKEY_LOCAL_MACHINE\SYSTEM\GWE\UserInput]
 "PromoteMouseInputToTouchInput"=dword:1

;--------- Ethernet Debug and Auto Launch manager -----------
IF BSP_AM33X_ETHDEBUG
#include      "$(_WINCEROOT)\PLATFORM\$(_TGTPLAT)\SRC\DRIVERS\ethDebugAutoRUN\ethDebugAutoRUN.reg"
ENDIF BSP_AM33X_ETHDEBUG

[HKEY_LOCAL_MACHINE\Init]
```

Build OS

- After all catalog items was modified as your requirement, save project and right click on ***eSOM335xOS*** in ***Solution Tab*** and click on Build eSOM335xOS.



- Build routine is a time-consuming process. When you build using the IDE, Platform Builder links and copies all necessary OS components, builds the platform, builds subprojects, copies files to the release directory, and makes the run-time image. After Build succeeded, navigate to below Folder:

WINCE700\OSDesigns\eSOM335xOS\RelDir\AM33X_BSP_ARMV7_Release

You should see ***NK.bin*** file created by Platform Builder. You can copy and replace this file with old ***NK.bin*** file and reboot your device with modified OS.

Restore Platform Builder to Factory Setting

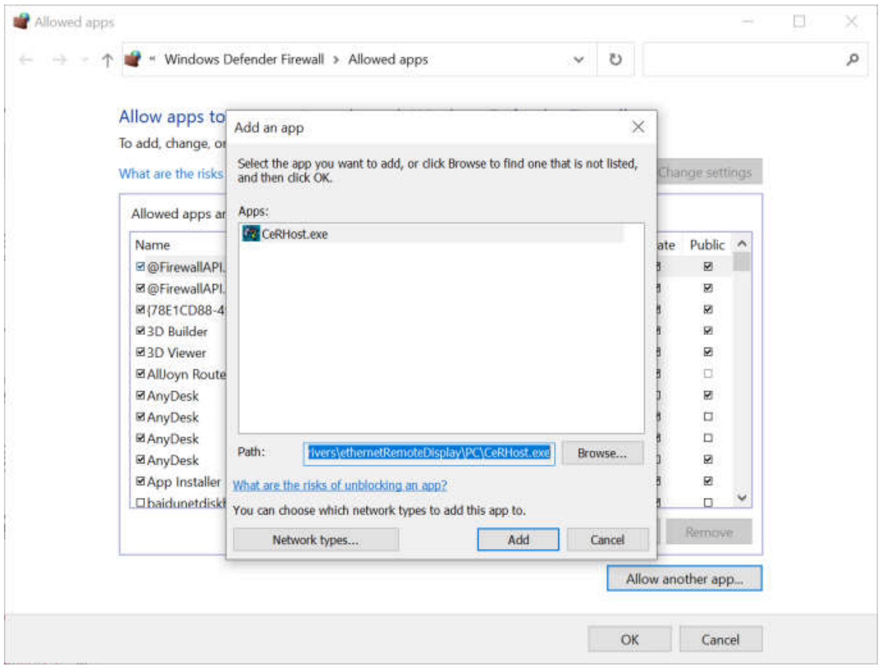
For restore item catalog settings to factory setting, eSOM335xOS.pbxml in ***eSOM335x\Source\WINCE700\OSDesigns\eSOM335xOS*** on your CD and paste and replace in ***Platform Builder installation Drive:\WINCE700\OSDesigns\eSOM335xOS***

OS Application

Remote Access

For access to your device remotely via ethernet cable, you can use ***cerdisp*** application, these applications exist in ***eSOM335x\ToolsDrivers\ethernetRemoteDisplay*** folder on your CD, follow these instructions to connect remotely to your device:

- **PC Side**: add ***CeRHost.exe*** app to Windows Firewall allowed apps, then Run ***CeRHost.exe*** as administrator.



Add an app

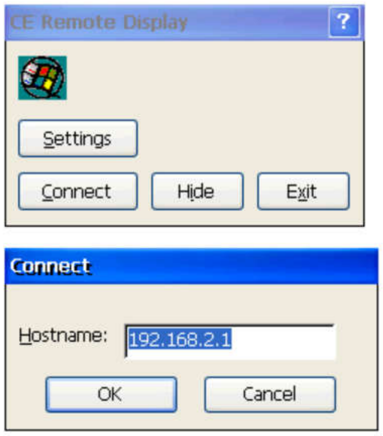
Select the app you want to add, or click Browse to find one that is not listed, and then click OK.

Apps: CeRHost.exe

Path: rivers\ethernetRemoteDisplay\PC\CeRHost.exe

You can choose which network types to add this app to.

- **Device Side**: Run ***CERDISP.exe*** click on Connect button, then in opened dialog box enter your PC IP:

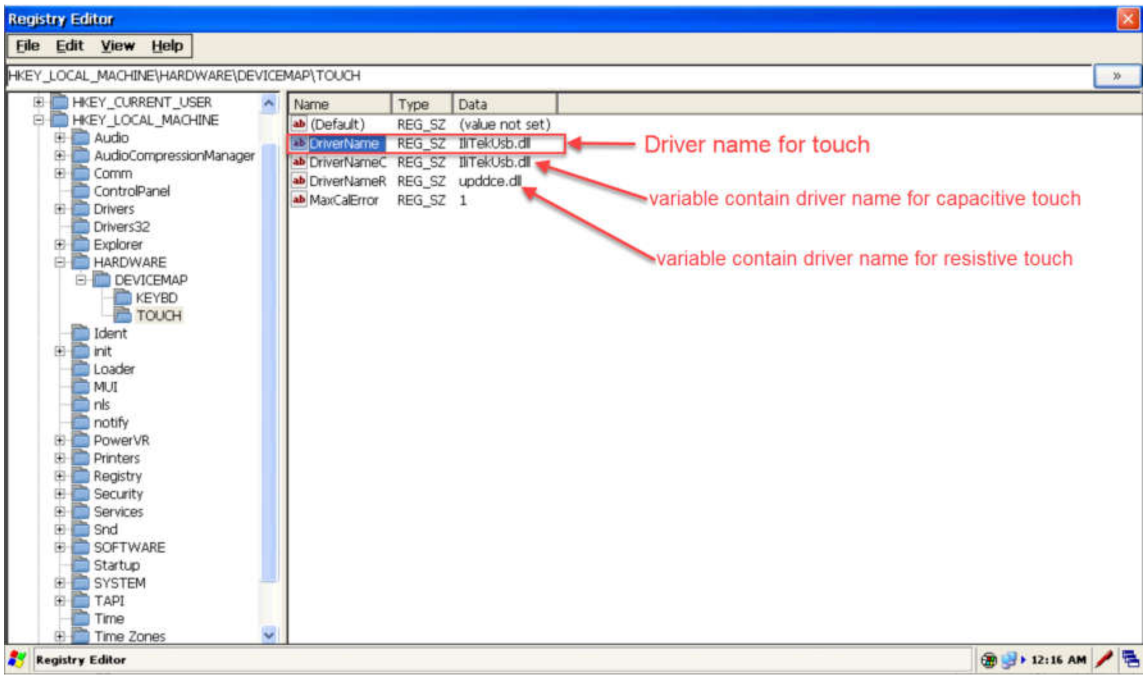


Hostname: 192.168.2.1

Note: In command prompt in your device navigate to ***cerdisp*** directory, then type:
cerdisp -h169.168.2.1 -c
you should type your host PC IP instead ***169.168.2.1***.

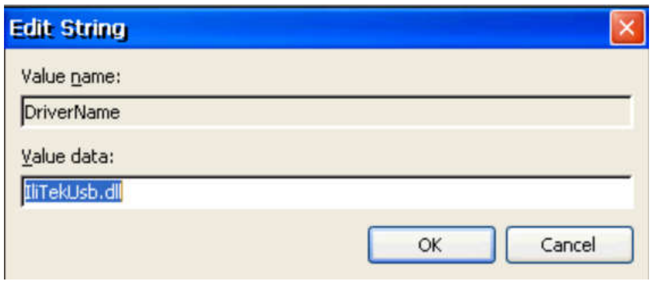
Touch Panel Setting

The eSOM335x support capacitive and resistive touch panel, by default capacitive type is selected for touch input, for switch to resistive touch open ***Registry editor*** from Windows folder and click on ***regedit***, then click ***HKEY_LOCAL_MACHINE*** then ***HARDWARE*** then ***TOUCH*** as shown in following figure:



Name	Type	Data
(Default)	REG_SZ	(value not set)
DriverName	REG_SZ	IliTekUsb.dll
DriverNameC	REG_SZ	IliTekUsb.dll
DriverNameR	REG_SZ	upddce.dll
MaxCalError	REG_SZ	1

Double click on ***DriverName*** in right page for determine DLL library for touch and write your driver's name in ***Value data*** box, by default this value filled by ***IliTekUsb.dll***, that shows capacitive touch is selected. For easy-to-use reason, two registries with ***DriverNameC*** and ***DriverNameR*** have been created to contain DLL library name of capacitive and resistive touch name respectively, therefore users can copy Value data and paste into the Value data of DriverName. After change value, restart your device, and device boots with your selected touch.



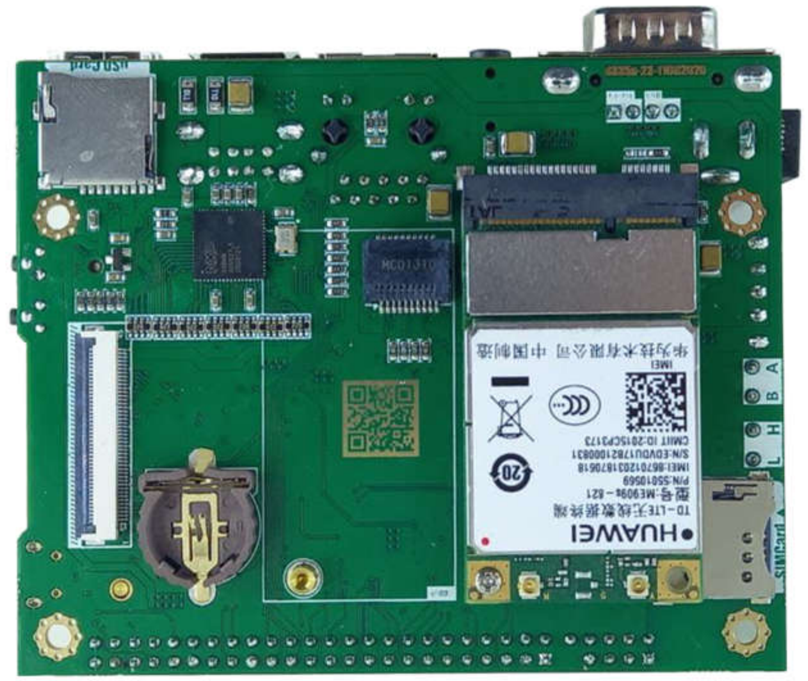
Value name: DriverName

Value data: IliTekUsb.dll

4G Module

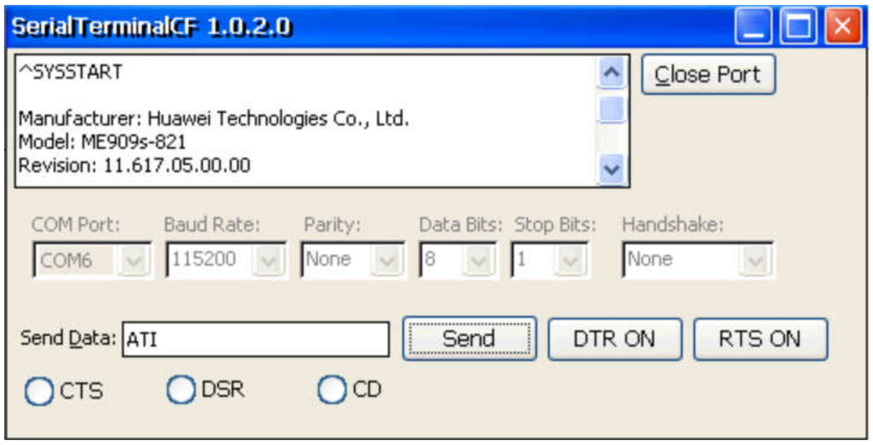
The eSOM335x support 4G module (***Model: HUAWEI ME909s-821***) for connect to the internet. In eDVK335x exits mPCIe slot that support USB only function and users can connect 4G module to this slot. For use this module follow below instructions:

- Plug 4G module to corresponding slot:



- Insert your sim card into corresponding slot.
- By default, COM6 is reserved for 4G module, user can test this module using a serial connection application like ***TerminalCF*** available in ***SampleCodes*** folder in net disk drive:

For example, send ATI command to module, then module should response some information about module.

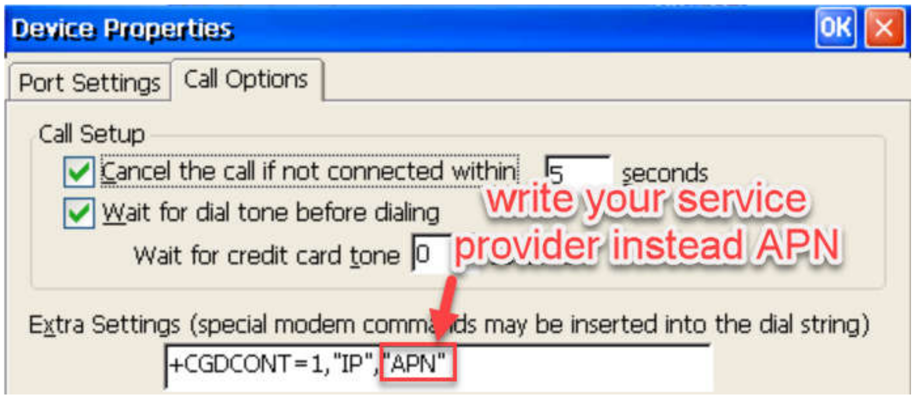


```
^SYSSTART

Manufacturer: Huawei Technologies Co., Ltd.
Model: ME909s-821
Revision: 11.617.05.00.00
```

COM Port: COM6 Baud Rate: 115200 Parity: None Data Bits: 8 Stop Bits: 1 Handshake: None

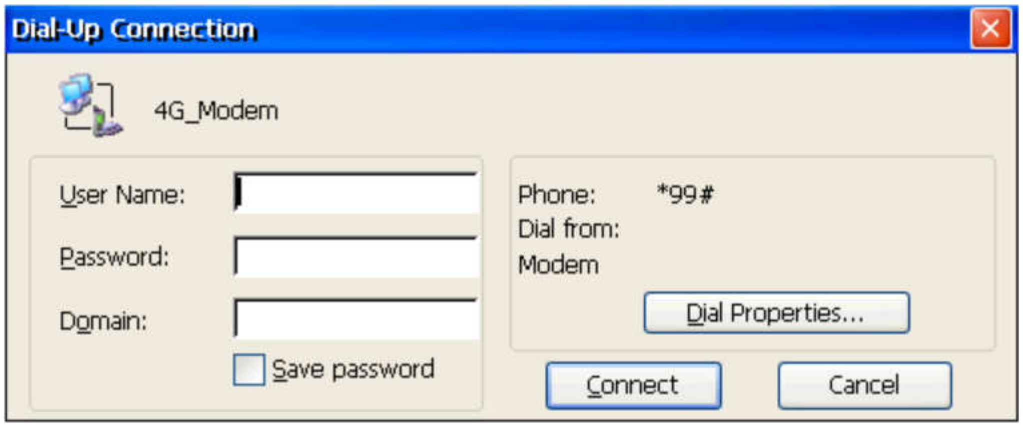
Send Data: ATI

- Click on close port and exit from TerminaCF, go to your device ***Control Panel*** and then click on ***Network and Dial-up Connections*** , then right click on ***4G_Modem*** and click on ***Properties***, in opened dialog click on ***Configure,*** a new dialog box is opened that users can config 4G module, most of setting have been done previously and users not need change any settings, only setting is service provider, for set this, go to ***Call Options*** tab and write your service provider instead of ***APN***, then click on OK.



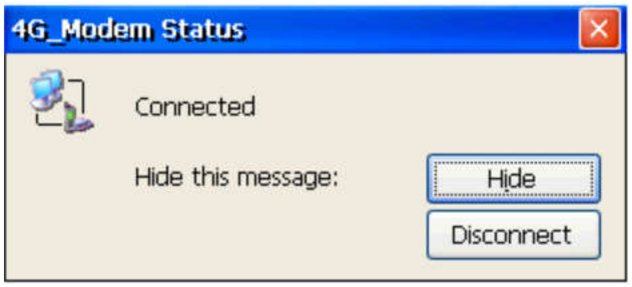
☑ Cancel the call if not connected within 5 seconds
☑ Wait for dial tone before dialing
Wait for credit card tone 0
Extra Settings (special modem commands may be inserted into the dial string)
+CGDCONT=1,"IP","APN"

- Double click on 4G-Module and, then click on ***Connect:***

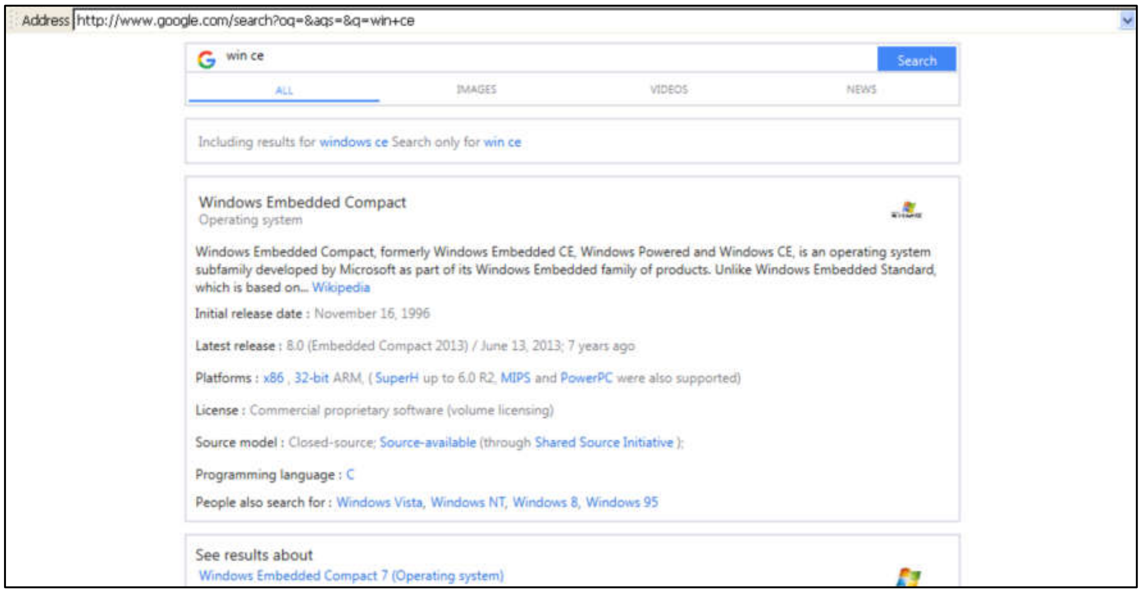


4G_Modem
User Name:
Password:
Domain:
☐ Save password
Phone: *99#
Dial from:
Modem

- If anything is OK, after authentication, modem status change to Connected. For test user can open internet explorer and search any things.



Connected
Hide this message:


Address http://www.google.com/search?oq=&aqs=&q=win+ce

Including results for windows ce Search only for win ce

Windows Embedded Compact
Operating system

Windows Embedded Compact, formerly Windows Embedded CE, Windows Powered and Windows CE, is an operating system subfamily developed by Microsoft as part of its Windows Embedded family of products. Unlike Windows Embedded Standard, which is based on... Wikipedia

Initial release date : November 16, 1996

Latest release : 8.0 (Embedded Compact 2013) / June 13, 2013; 7 years ago

Platforms : x86 , 32-bit ARM, (SuperH up to 6.0 R2, MIPS and PowerPC were also supported)

License : Commercial proprietary software (volume licensing)

Source model : Closed-source; Source-available (through Shared Source Initiative);

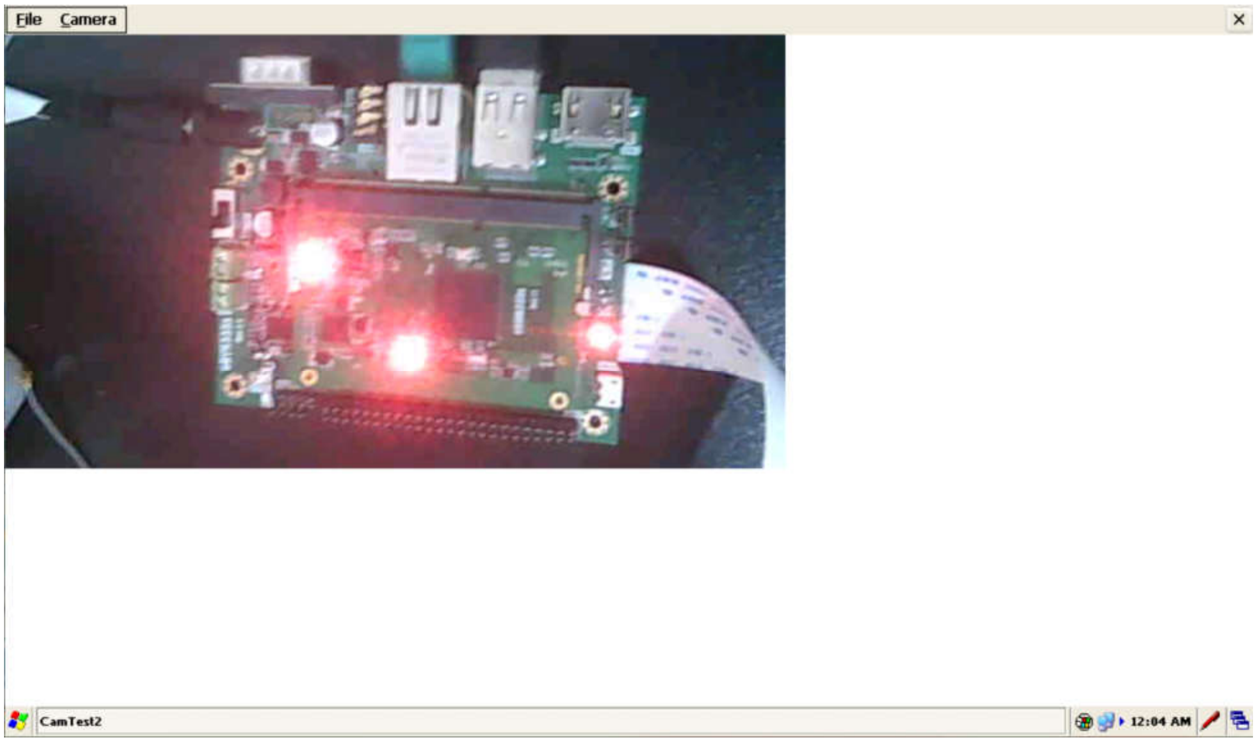
Programming language : C

People also search for : Windows Vista, Windows NT, Windows 8, Windows 95

See results about
Windows Embedded Compact 7 (Operating system)

Camera

The eSOM335x support USB camera (A Logitech USB Camera was tested), connect this camera to USB port of carrier board. An application exists in Windows folder named ***CamTest*** for test and use of this camera, after running it, you can see camera output on your screen.



Useful links

https://www.microsoft.com/en-us/download/details.aspx?id=28757

https://www.hpcfactor.com/downloads/