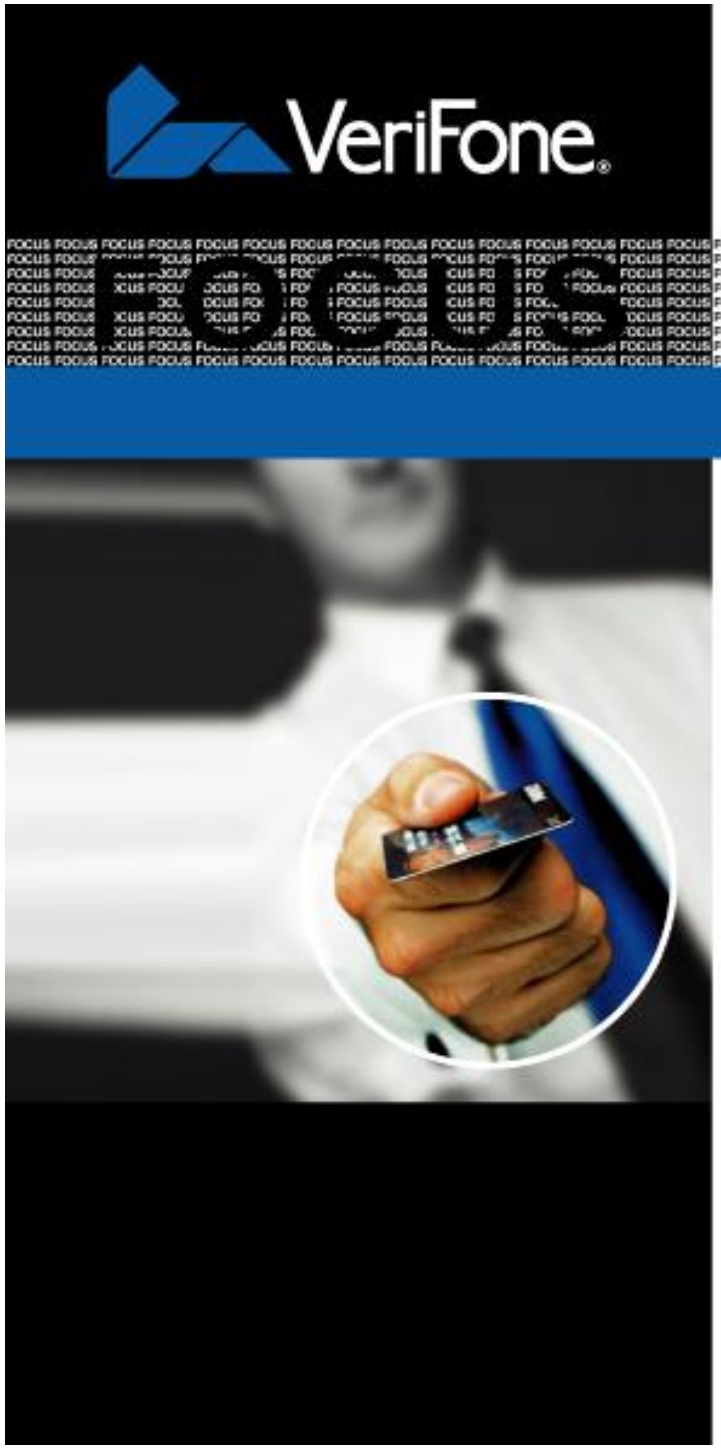
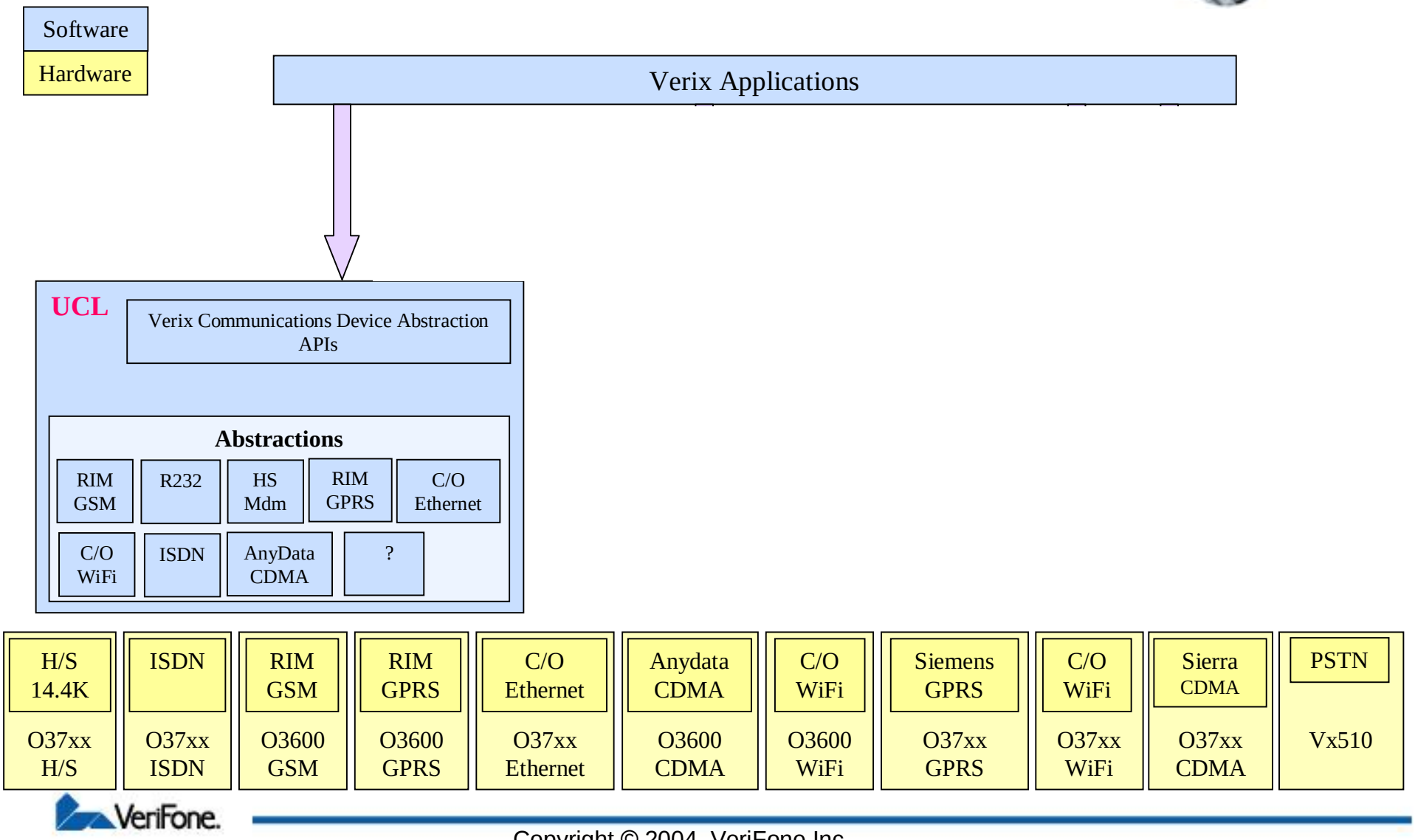


UCL

Universal Communication Library

Universal Communication Library



Introduction

- Common APIs
- Easy switch over to different media
- Device specific details encapsulated
- Simultaneous communication
- Critical parameters are application provided
- Configurable timeouts

UCL Architecture

- Object Oriented
- ‘C’ Interface
- One class per media
- Configurable timer.

UCL Architecture

- Uses Factory design pattern
- CUCLFactory
  - Creates appropriate class objects
  - Implements SmartLink feature
- CUCL
  - Base class for all media
  - Implements common functionality

UCL Architecture

- CCommTimer
  - Virtual class for timer implementation
  - Defines the timer interface
- CVerixCommTimer
  - Default implementation for Timer
  - Application can have their own using CCommTimer

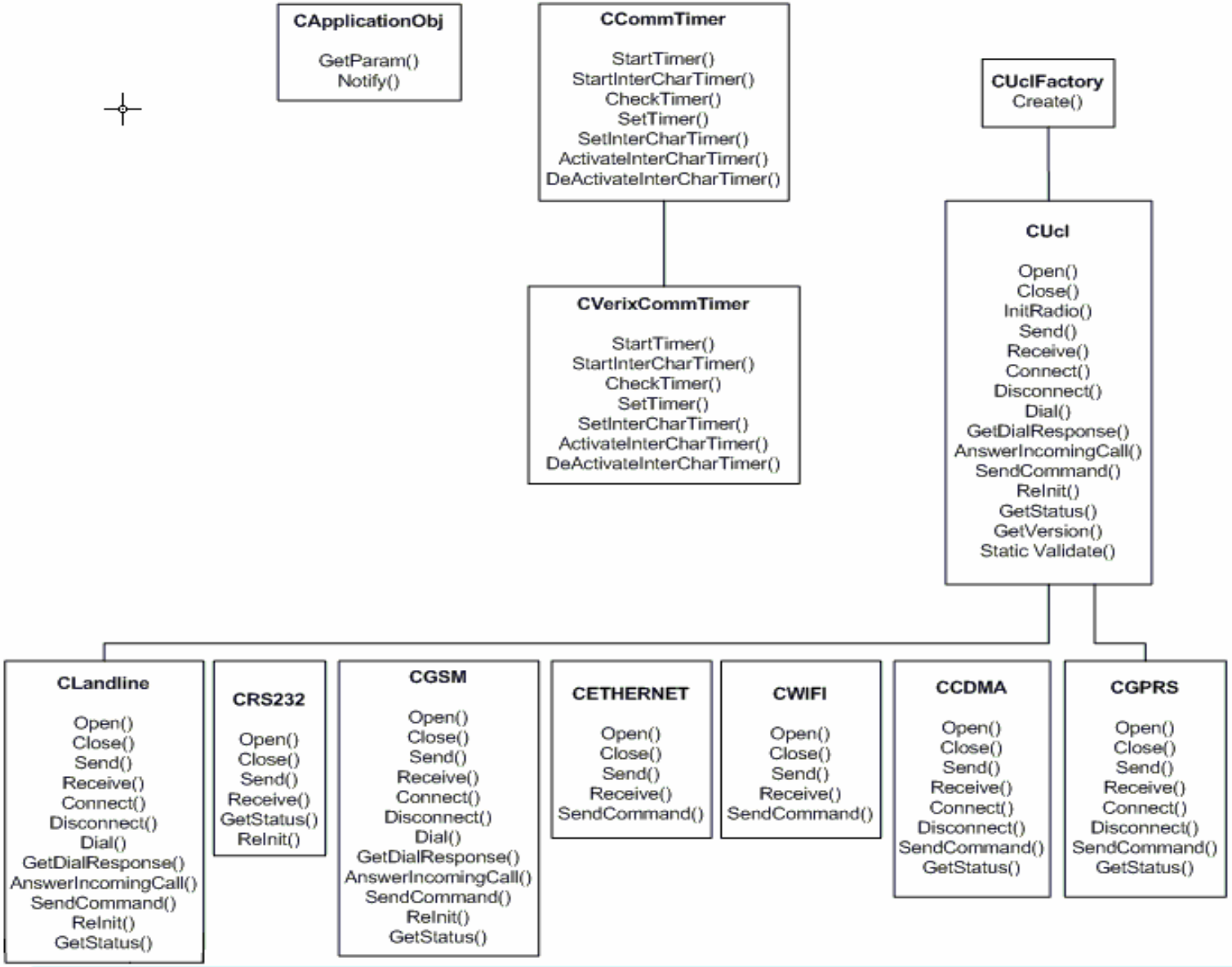
UCL Architecture: CCommTimer

- APIs
  - StartTimer()
  - StartInterCharTimer()
  - CheckTimer()
  - SetTimer()
  - SetInterCharTimer()
  - ActivateInterCharTimer()
  - DeactivateInterCharTimer()

UCL Architecture

- CApplicationObj
  - Interface to the application
  - Implements GetParam()
  - Implements Notify()
  - Way to get application inputs

UCL Architecture



UCL Architecture: CUcl

- APIs
  - Open()
  - Close()
  - ReInit()
  - SendCommand()
  - Connect()
  - Dial()
  - GetDialResponse()
  - Disconnect()
  - Send()
  - Receive()
  - AnswerIncomingCall()
  - GetStatus()
  - GetVersion()

UCL: Supported Media

- LandLine
- RS232
- CDMA
- GPRS
- GSM
- Ethernet
- WiFi

UCL: Supported Media

- Omni37xx CDMA
- Omni37xx GPRS
- Omni37xx WiFi
- Omni 37xx Ethernet
- Omni 3600 CDMA
- Omni 3600 WiFi
- Omni 3600 GPRS

UCL: Supported Media

- Vx610 CDMA
- Vx610 GPRS
- Vx610 WiFi

UCL: Header Files

- #include <uclfactory.h>
  - All ucl interface definitions
- #include <tcpinterface.h>
  - Interface to TCPIP
- #include <verixtimer.h>
  - Sample timer implementation

UCL: Binaries

- Debug version
  - Ucl.a (Output\SDS\Files\Static\Debug) – for Verix
  - Ucl.a (Output\RV\Files\Static\Debug) – for Verix V
  - Log enabled
  - Bigger than release version
- Release version
  - Ucl.a (Output\SDS\Files\Static\release) – for Verix
  - Ucl.a (Output\RV\Files\Static\Debug) – for Verix V
  - Log disabled
  - Smaller in size

UCL Manuals

- Available in docs
  - UCL Programmer's Guide.pdf
  - UClHelp.chm

UCL: API Sequence

- uclFactory = CreateUclFactory();
- timer = (CommTimer*)CreateVerixCommTimer();
- comnObj = uclFactory->Create(COM3, &retVal, (ApplicationObj *)AppObject,GPRS); // for Verix
- comnObj = uclFactory->Create(COM2, &retVal, (ApplicationObj *)AppObject,GPRS); // for Verix V
- comnObj->Open(comnObj,timer);

UCL: Configuration

- Timeouts
  - SetTimer()
- Parameters
  - GetParam()
  - BAUDRATE,FORMAT,PHONE_NUMBER,INIT_STRING,HANGUP_STRING,MODEL_NO,TERMINAL_TYPE, etc.,
  - Default
    - DEFAULT as the value for GetParam()
  - InitRadio() to paralalyze the activity

UCL: Features

- GetStatus()
  - PORT_STATUS
  - DEVICE_STATUS
  - DOCK_STATUS
  - PDPCONTEXT_STATUS
  - REG_STATUS
  - BATTERY_STATUS
  - SIGNAL_STATUS
  - COVERAGE_STATUS
  - LINE_STATUS
- Notify()
  - Use to update the status of PPP
  - Docked and undocked

UCL: Features

- Predial
  - Dial()
  - GetDialResponse()
- Change the baud
  - ReInit()
- Send raw command
  - SendCommand()
- Same resource file for all terminals
- External modem on COM1
- ASYNC and SYNC communication

UCL: Media Differences

- The resource file
  - UCL maintains the media details in .res file
  - Media specific files needs to be downloaded
- GetParam()
  - MODEL_NO
  - TERMINAL_TYPE
- Timeouts

UCL: Media Independent Code

- Store UCL inputs in a data file
  - TXO file
  - GetParam() is used by UCL for inputs
- Use SVC_INFO_MODULE_ID()
- Load the appropriate data file at runtime

UCL: SmartLink

- UCL is based on Objects
- Only the object code of the used media is linked
- SmartLink at compile time

UCL: Advantages

- Device dependent details abstracted
- Device parameters have DEFAULT
- Provision for application to change default
- Easy to switch among devices

UCL Summary

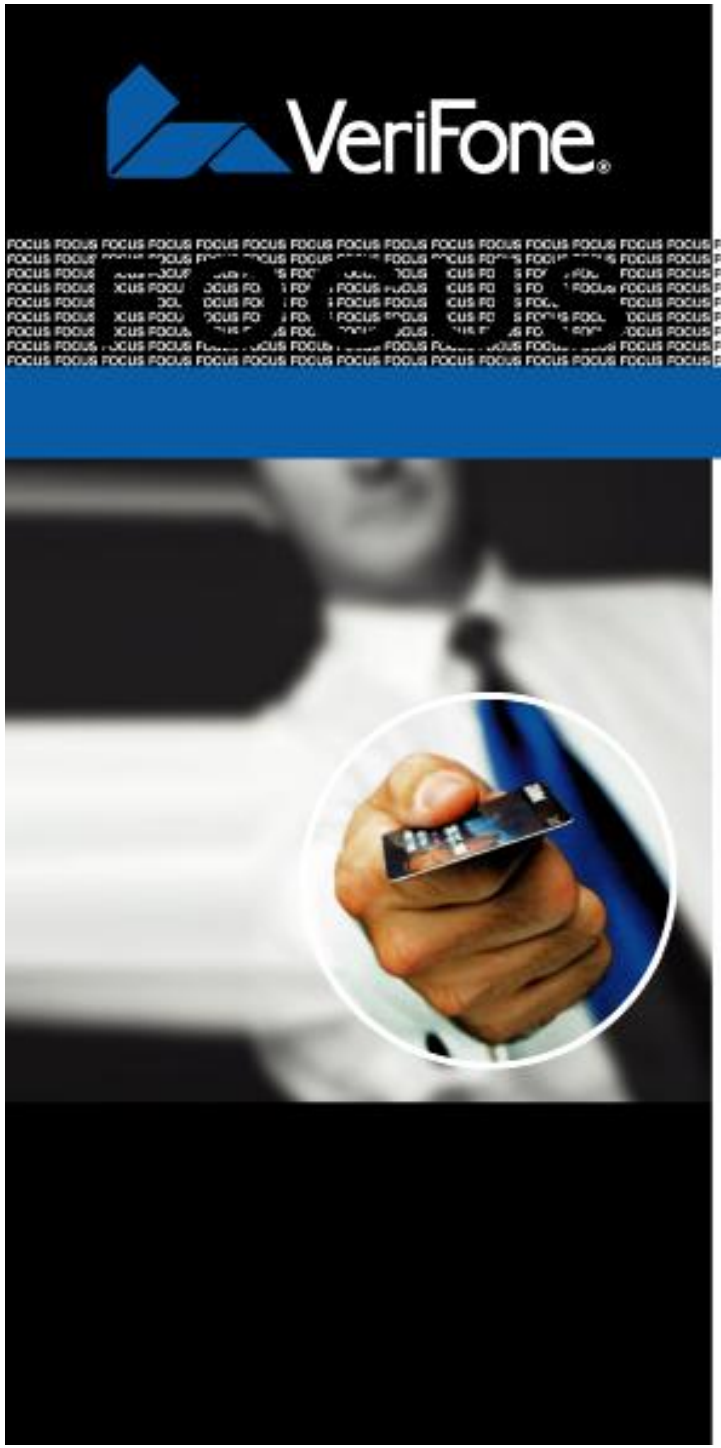
- UCL = Universal Communications Library
- Provides communication device abstraction for applications
  - Normalized APIs enable device change without application code change
  - Facilitates plugging in application-specific communication devices without affecting the functionality of the devices supported by the library
- “Smart Linking” feature delivers memory efficiency
  - Linking the application to only the devices used by application
- Application can implement its own timeout functionality

UCL Summary

- Supports Omni 3750 wireless IP communication devices:
  - GPRS (Siemens MC55/56)
  - CDMA (Sierra EM340)
  - WiFi (ConnectOne CO561)
- Supports Vx610 wireless IP communication devices:
  - GPRS (Siemens MC55/56)
  - CDMA (Sierra EM340)
  - WiFi (ConnectOne CO561)

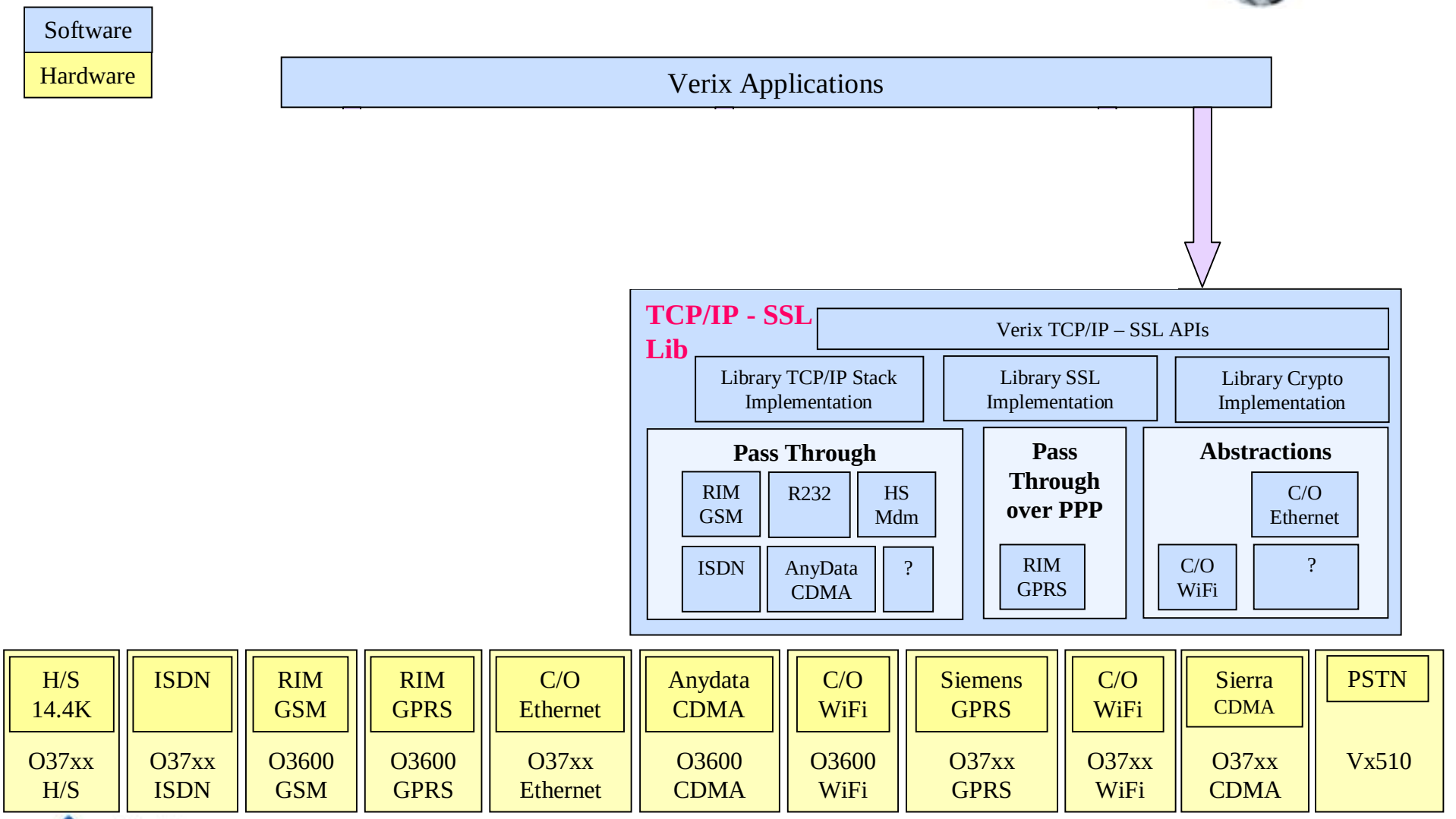
UCL Summary

- Facilitates Battery Power management, connection loss management and signal quality management on Vx610
- Dial back up using High speed modem used in Omni3750 Combo I/O modules and Vx610 base unit
- Other Devices Supported:
  - GSM (RIM, GM25 Ericssion on Omni3600)
  - CDMA (AnyData.net on Omni3600)
  - WiFi (ConnectOne C0561 on Omni 3600)
  - Landline on Omni3300,3600,3750, Vx510, Vx610



TCP/IP Library

TCP/IP Library



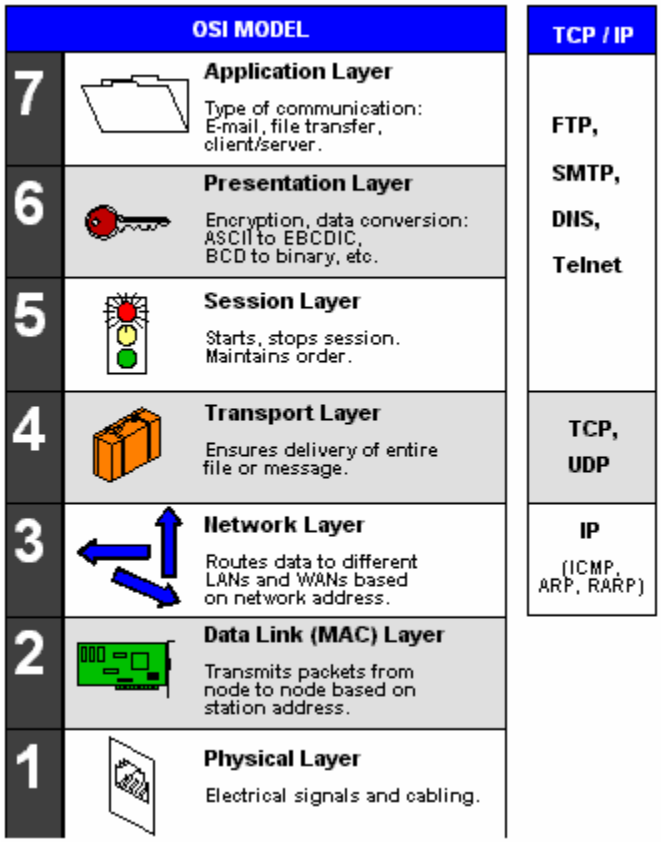
TCP/IP: What is TCP/IP?

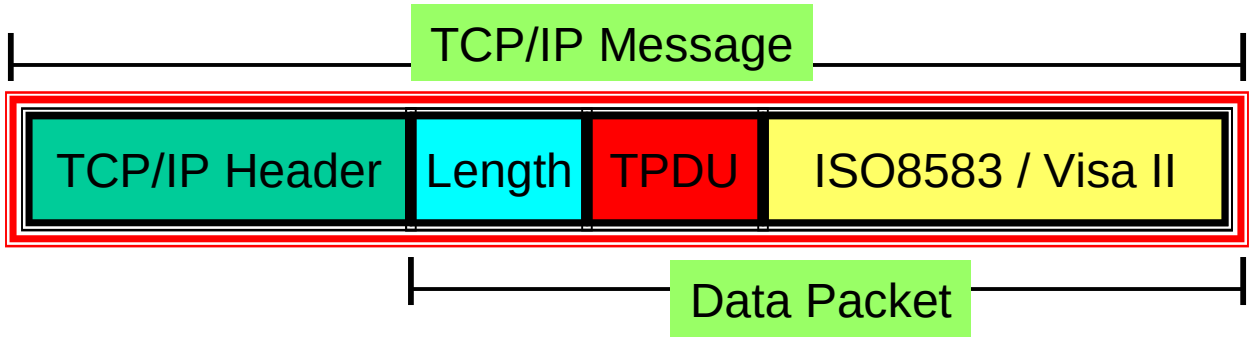
- TCP/IP consists of a set of protocols which will be used in different facets of communication
- TCP is a sophisticated, reliable, byte-stream protocol
  - It is responsible for verifying the correct delivery of data
  - TCP adds support to detect errors or lost data and to trigger retransmission until the data is correctly and completely received
- IP is responsible for moving packet of data from node to node
  - IP forwards each packet based on a four byte destination address
  - IP operates on gateway machines that move data from department to organization to region and then around the world.

TCP/IP: OSI vs TCP/IP

- Layered architecture
- OSI Protocol stack
- TCP/IP Protocol stack
- OSI vs. TCP/IP protocol stack

OSI Model and TCP/IP Transactions





Example:

Header: 20 Bytes

POS IP and Port

Host IP and Port

TCP/IP Control Info

Message Length: 2 Bytes

TPDU: 5 Bytes

Message: N Bytes

TCP/IP: TCP/IP Protocol Stack Layer

- Datalink layer / Physical layer
- Internetwork layer
- Transport layer
- Application layer

TCP/IP: Verix TCP/IP Library

- What is Verix TCP/IP Library?
- Protocols supported by Verix TCP/IP stack
- Components of Verix TCP/IP Library
  - IP Address conversion API
  - Host Information API
  - Network configuration API
  - Socket API
  - Conformance to BSD Standard
- Conventions used in Verix TCP/IP Library
  - Success and failure codes
  - When to look for the value of errno variable?
  - Introducing the participants to Programmer's guide.

TCP/IP Library: Header Files

- Host interface APIS
  - Netsetup.h
- Socket APIS
  - TCPIP.h
  - Vsocket.h
  - Sslinit.h
- Host information APIS
  - Inet.h
- Timer APIs
  - Verixtimer.h
- Application object
  - Appobj.h

TCP/IP Library: Binaries

- Verix
  - <TCPIP>\Output\SDS\Files\Static\Debug
    - Tcpip.a
    - Ssl.a
    - Crypto.a
- Verix V
  - <TCPIP>\Output\RV\Files\Static\Debug
    - Tcpip.a
    - Ssl.a
    - Crypto.a

TCP/IP Library: Application Defined Data Link Layer

- Application can define TCPIP's data link layer
  - UCL
  - Any other datalink layer
- UCL
  - short AssignUclToTCP(UCL *ucl, CommTimer *timer, unsigned long openTimeout, unsigned long connectTimeout, unsigned long disconnectTimeout, unsigned long closeTimeout, unsigned long sendTimeout, unsigned long receiveTimeout);
- Any other datalink layer
  - assignDLtoTCP(short (*appopen)(void), short (*appconnect)(void), short (*appdisconnect)(void), short (*appclose)(void), short (*appsend) (unsigned char * sendBuffer , short bytesToSend, unsigned long timeout), short (*appreceive)(unsigned char * receiveBuffer, short bytesToReceive, unsigned long timeout));

TCP/IP Library: Application Defined DL

- APIs
  - appOpen
  - appConnect
  - appDisconnect
  - appClose
  - appSend
  - appReceive
- To continue operation return >= ‘0’
- To cancel the operation return < ‘0’
  - Use to cancel the PPP operation, in case of PPP failure

TCP/IP Library: API Sequence

- uclFactory = CreateUclFactory();
- timer = (CommTimer*)CreateVerixCommTimer();
- comnObj = uclFactory->Create(COM3, &retVal, (ApplicationObj *)AppObject,GPRS); // COM2 for Vx610
- AppObj appObj; fill this structure
- AssignUcltoTCP()
- Fill netparams structure
- Netconfig()
- Setparam()
- vSSL_Init() if SSL required
- Netconnect()

TCP/IP Library: IP Address Conversion

- Dotted-decimal notation for IPv4 addresses
- Converting an IP address in dotted-decimal notation to binary
  - inet_aton() function
  - inet_addr() function
- Converting a binary IPv4 address to a string in dotted-decimal notation
  - inet_ntoa() function

TCP/IP Library: Host Information API

- Introduction to DNS
  - What is DNS?
  - Why DNS?
- How is DNS information maintained by the TCP/IP Library?
- Adding a host record
  - addhost() function
- Retrieving host record
  - hostent structure
  - Using name of the host : gethostbyname() function
  - Using the IP address of the host : gethostbyaddr() function

TCP/IP Library: Host Information API

- Deleting a host record
  - deletehost() function
- Getting the count of host records
  - gethostcount() function

TCP/IP Library: Network Configuration API

- What is a network interface?
- PPP connection to Internet Service Provider
  - What is PPP Connection?
  - Protocols involved in a PPP connection
  - Static and Dynamic IP address assignment
- Defining a network interface
  - netconfig() function
- Retrieving network interface details
  - getnetconfig() function

TCP/IP Library: Network Configuration API

- Connecting to Internet Service Provider using PPP
  - netconnect() function
- Terminating PPP connection
  - netdisconnect() function
- How to find the status of a PPP connection?
  - netstatus() function
  - Why is it required to find the status of PPP connection?
    - Blocking and non-blocking calls
    - ISP timeout
    - Lan link lost detection
    - SIM not configured (GPRS/CDMA)
- Network interfaces supported by the TCP/IP Library
  - One network interface per application

TCP/IP Library: Static vs. DHCP IP

- net_param.ipAddress
  - Specify the IP address to use
  - Set to 0.0.0.0 to use DHCP
  - Final IP is decided by the ISP or DHCP server

TCP/IP Library: TCP Sockets

- Creating a socket
  - socket() function
- Binding an IP Address and port number to a socket
  - sockaddr and sockaddr_in structures
  - bind() function
  - significance of bind() for a TCP server socket
  - significance of bind() for multi-homed hosts
- Connecting to a TCP server socket
  - What is a TCP connection?
  - Connect() function

TCP/IP Library: TCP Sockets

- Converting a socket to server socket
  - listen() function
- Waiting for a TCP client to connect to TCP server socket
  - accept() function
- Writing data to a TCP socket
  - send() functions
  - Define send timeout using SO_SNDTIMEO option on setsockopt() function

TCP/IP Library: TCP Sockets

- Reading data from a TCP socket
  - recv() functions
  - Define receive timeout using SO_RCVTIMEO option on setsockopt() function
- Terminating a TCP connection
  - closesocket() function
  - What is the effect of ‘linger time’ on closesocket?
    - Linger structure
    - How to set linger time for a socket?
      - SO_LINGER option
  - SO_CLOSEDELAY socket option using setsockopt() function

TCP/IP Library: TCP Sockets

- Keeping a TCP connection alive if the connection is idle for longer time.
  - Setting SO_KEEPALIVE option on a socket using setsockopt() function
  - Keepalive packets
- TCP client-server model
  - Concurrent server model and iterative server model

TCP/IP Library: UDP Sockets

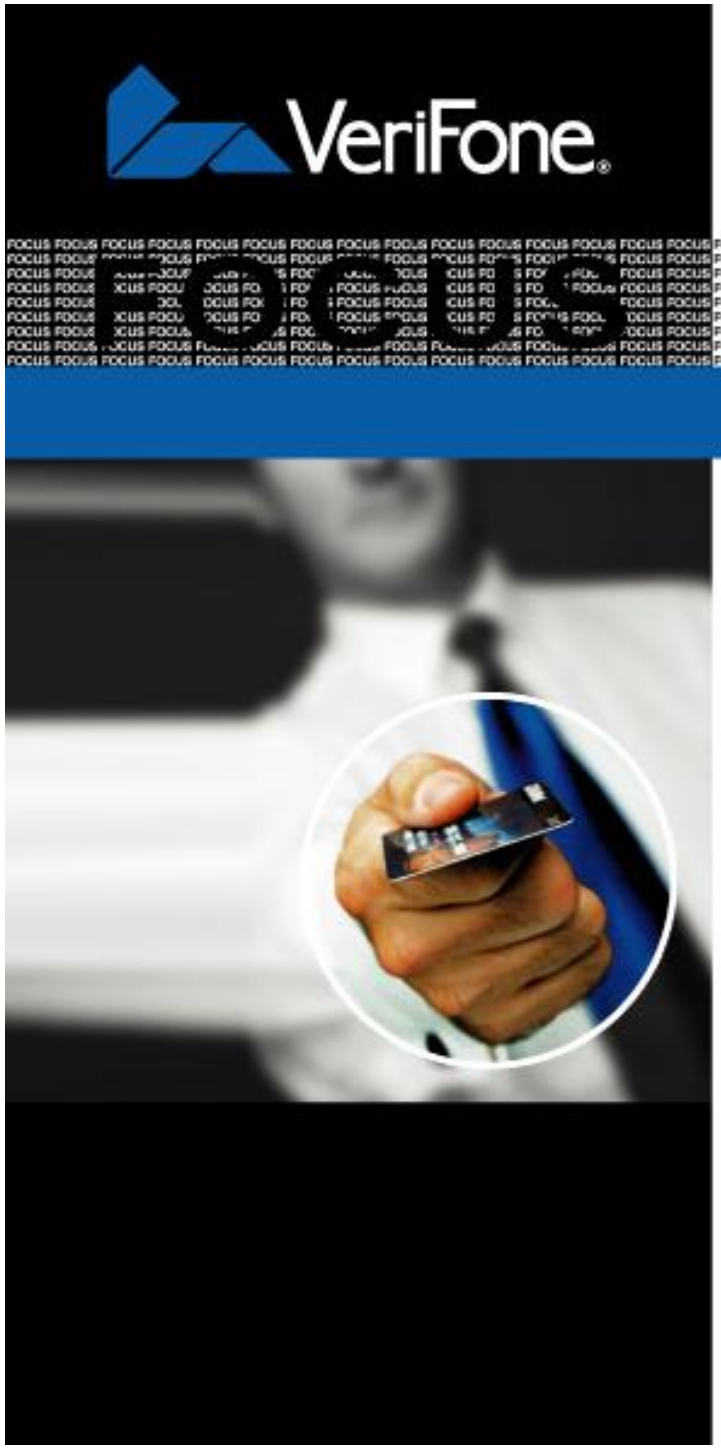
- UDP client-server model
- Creating a UDP socket
- Writing data to a UDP socket
  - sendto() function
- Receiving data from UDP socket
  - recvfrom() function
- Concept of UDP connection
  - connect() function on a UDP socket
  - send(), writesocket() functions
  - recv(), readsocket() functions.

TCP/IP Library: I/O Models

- Blocking vs. Non-blocking sockets
  - recv(), recvfrom(), readsocket() functions
  - send(), sendto() and writesocket() functions
  - accept() function
  - How to convert a socket to non-blocking and vice-versa?
    - ioctlsocket() and fcntlsocket() functions
  - Non-blocking sockets and yield() function

TCP/IP Library: Verix vs External Stack

- Media
  - LandLine, CDMA, GPRS, GSM are Verix stack
  - Ethernet and WiFi are External stack
    - ConnectOne stack
- Interface
  - Same API interface for both
  - Library provides transparency
  - net_param.datalinkProtocol differentiates the stack
    - DL_PPP_CO_ETH
    - DL_PPP_APP_DEFINED



Programming Notes

Programming Notes

- UCL as datalink layer
  - To support future media
- Timeouts
  - SetTimeout(SEND_TIMEOUT,1000);
  - SetTimeout(RECEIVE_TIMEOUT,3000);

Programming Notes

- Check the return values & errno after calling each of the functions
- Take appropriate steps
  - -501 if no IP assigned
    - Reset iChip by sending BREAK signal
  - -107 No response or invalid from iChip
  - -717 baud rate mismatch
    - Enter the iChip to auto baud
  - -174 Link lost situation
    - Retry the operation
- Dial-backup
  - -501 – Reset the IP address.
  - Wait required to switch to Wireless from dial backup

Programming Notes: Connect One

- setparam
  - Ethernet
    - WAIT_ICHIPRESET
      - Wait time for CO soft reset
    - TCP_TIMEOUT
      - Change the TTO timeout
  - WiFi
    - WIFI_CHANNEL
    - WIFI_SYSID
    - WEP_MODE
    - WEP_KEYINDEX
    - WEP_KEYVALUE

Programming Notes: WiFi

- Setting up Access point
  - SSID
  - Tuning channel
    - Terminal and AP should use the same channel
  - Secure communication between terminal and AP
    - Enable WEP mode in both AP and terminal
      - 64bit and 128 bit
    - WEP key should match between AP and terminal