Fax Technology for Embedded Systems

Michael Chase
COMtrix Systems, Inc.
Boulder, Colo.

Michael J. Chase is on the graduate school faculty at the University of Colorado, where he teaches telecommunications and programming languages. He has worked at AT&T Bell Laboratories as a member of the technical staff, and at IBM's information products division. Chase professionally teaches C and C++ seminars for customers such as AT&T, IBM, Ameritech, Los Alamos National Labs, and Digital Equipment Corporation; he is also a principal of COMtrix Systems, Inc., which designs and manufactures wireless embedded system communication hardware and software for the process control industry. He holds a BSEE from Columbia University, an MSCS from the University of Southern California, and an MSEE/Telecommunications from the University of Colorado.
ABSTRACT

Data communication options for embedded system designers are expanding at exponential rates. One low cost option for transferring information is the use of off-the-shelf fax modem technology. Small, reliable, and economical (less than $100.00) single part fax-modem/DAA packages are available from several sources and can be called upon to send text and graphics to the most ubiquitous printing device in the world — the Group 3 fax. With some accommodation, the same technology can be used to send half-duplex data at 9.6 and 14.4 kbps from embedded systems.

INTRODUCTION

The motivation for an embedded system designer to examine how a fax works and what hardware and software is involved is twofold:

1. Fax capabilities can augment an embedded system with the ability to send reports to the most ubiquitous printing device in the world - the G3 facsimile machine which is virtually available in every office and can be reached from anywhere in the terrestrial PSTN (Public Switched Telephone Network) as well as through wireless cellular networks.

2. Inexpensive fax modems can be utilized to establish full duplex 2.4 kbps and half duplex 9.6 kbps data links to/from embedded platforms. Also, an understanding of fax modem protocols will prepare the designer for the forthcoming CCITT T.434 BFT (Binary File Transfer) protocol that is specifically designed to accommodate error checked block transfers.

There are literally hundreds of fax boards and just as many application programs to run them. If, however, your embedded platform is MS-DOS/ISA/EISA based, this glut of fax hardware and software products is essentially unusable. In fact, most manufacturers of commercial fax software make assumptions that are often unrealistic for an embedded controller. These minimum assumptions often include:

1. One megabyte MS-DOS file system.
2. 16 MHz 80X86 ISA bus.
3. 256K of RAM available.
4. Pre-defined (8259) interrupt controller scheme.
5. ROM BIOS facilities available.

Figure 1

Embedded systems designers generally must contend with limited RAM/ROM, no file system, and real time constraints placed on the processor's cycles (occupancy). Due to
limited resources on 8-bit platforms (time, space, cost), assembly language is often used. The C language is popular on more modern 16-bit and 32-bit platforms. A standardized set of kernel services is usually available, but is non-standard across most processor instruction sets².

For the purposes of this discussion we will assume that the “general” embedded system has at least the following aggregate attributes:

1. Cooperative or preemptive kernel with basic task scheduling and memory allocation primitives.
2. 2.64K bytes of RAM/ROM address space.
3. Ability to perform serial I/O.
4. Processor cycles on a time-available basis; current processor occupancy is such that bursty interrupt traffic can be accommodated.

This paper identifies the design issues and suggests candidate solutions for those who desire to craft fax capabilities into an embedded system.

HOW DOES THE G3 (GROUP 3) FAX WORK?

The process of faxing a piece of paper is straightforward. Scan and digitize a surface image, encode and transmit the digital information, with the received digital information, decode and reconstruct the surface image on paper using thermal or xerographic printing techniques. Not so simple are the CCITT³ and EIA/TIA⁴ standards to which the process must conform, provisions for backward compatibility to existing generations of fax technology, and the resolution of compatibility issues when manufacturers make “assumptions” about standards that are not specific enough or are ambiguous. Many of these issues are attended to by fax modem manufacturers, however, everyone does so differently enough to cause compatibility problems. An obstacle facing embedded systems designers is that the standards making process is slow, in constant flux, and is sometimes ignored by fax modem manufacturers.

There are essentially four (4) generations, or groups, of fax technology: G1, G2, G3, and G4. The primary focus of this discussion will be on G3 fax hardware and protocols that utilize the PSTN. A G4 fax utilizes high speed digital transmission facilities such as DDS (56 kbps), T1 (DS0 (56/64 kbps, DS1 1.44 Mbps), and ISDN BRI (2B+D; 2*64 kbps + 16 kbps). G3 is much more ubiquitous and much less expensive than G4, at least today (/YY).

The fax machine sitting on your desk is already a very complex embedded system responsible for photo digitizing, stepper-motor control, LED and LCD displays, keypad decoding, sound generation, and interfacing to the PSTN in both voice and data modes. Clearly, we do not desire to implement all that. So then, what has to be done and how hard is it to integrate fax functionality into an existing embedded system that, for example, dispenses soda cans for money? It’s straightforward if you make an informed selection from the fax subsystems available in today’s market.
Fax modems have been available for quite some time and they are as simple to use as standard data modems with "AT" commands. However, not until recently (YY) has the EIA/TIA officially endorsed a standardized set of high level "AT" commands by which manufacturers can build, and the embedded system designer can rely on for a high degree of universal functionality. The recent EIA/TIA 592 (Class-2) standard is of particular interest because it provides the highest degree of functionality and is therefore the easiest to program and integrate. Before EIA/TIA 592 or any other standards are discussed, the mechanics of a fax transmission are examined.

Encoding the Scanline

When sending a fax, the digital image of a page (from top to bottom) is encoded by performing line-at-a-time scanning in left-to-right order. A typical page 8.5" by 11.0" is divided into 1080 scan lines. Each scan line has 1728 pels (picture elements). Capabilities are usually provided to divide the page into 2160 scan lines in order to send a high resolution image.

![Figure 2](image)

Each scan line is traversed to identify runs of black and white pels, identifying the length of each black or white pel run, and selecting a pre-defined Huffman code sequence (bit pattern) that represents the run (see figure 3). The most common (probable) run lengths have been assigned short codes and the least common run lengths have been assigned longer codes. The concatenation of these Huffman codes essentially becomes the synchronous information bit stream transmitted on an established data link to a remote fax machine. Once runs of black and white for a given line are sent, the next line is then processed until there are no more lines on a page to send.
When a line is complete, a special EOL (End Of Line) pattern is sent (see figure 4). When a page is complete, six EOL marks are sent to mark the end of a page and the two fax machines then return to a control negotiating mode to accommodate error correction and/or further pages. As a typical low end paper fax machine ingests a page of paper it generally scans, encodes, and transmits binary data a scan line-at-a-time.

The timing shown in figure 4 is such that T is the minimum amount of time that must elapse for the transmission of a scan line. If the time needed to send SCANLINE DATA is less than T, fill bits (0’s) are sent. T can vary between 0 and 40 ms to accommodate a mechanical latency of the printing apparatus of the receiving fax machine. T plays an important role in the design of serial communications software, especially at the interrupt level. The data is sent at 9600 bps in synchronous fashion — no start or stop bits, no “slack” time between bits.
At this point you may be asking: Who determined the Huffman codes? How is the telephone connection established? What modulation scheme is used in data transmission over the PSTN? How are the capabilities of the remote fax machine determined? What flow control and timing schemes are used? etc. All of these questions are answered by multiple standards and recommendations provided by both the CCITT and EIA/TIA. To simplify the investigation of these standards, we must lay down criteria for the selection of the fax subsystem that we desire to use in an embedded system. The embedded fax subsystem criteria are:

1. be able to send text (and optional graphic images) directly from an application program to a remote G3 fax.
2. be able to establish a full duplex 2400 bps data connection using standard "AT" modem commands.
3. minimize the amount of additional power and PCB space needed.
4. cost less than $100.00 (Qty. 25).
5. maximize the high level functional interface to simplify software integration.
6. adhere to as many sanctioned standards (CCITT, EIA/TIA) as is practical.
7. minimize the demands that it places on the processor occupancy and memory of the embedded platform.

Figure 5

We now proceed to identify specific CCITT and EIA/TIA standards that participate in meeting the goals listed in figure 5. The hardware, software, and firmware combination is chosen so that the embedded systems developer has a minimal amount of work to integrate fax capability to an existing system. In larger quantities (100+), there may be cost trade-offs and economies that would alter the configuration suggested. The EIA/TIA Standard 592 (Class-2) is chosen, as it most closely meets the goals of the design — it is high level and has the fax subsystem do almost all the work of sending a fax. Multiple fax modem manufacturers provide EIA/TIA 592 functionality. It is acknowledged that EIA/TIA Standard 578 (Class-1) exists and is usable; Class-2 is chosen as it further eases the responsibilities of the host application.

Figure 6 shows a typical topology of a Class-2 fax modem subsystem. There are four major components in this subsystem: 1) DSP-Controller/RAM/ROM. 2) Modem D/A, A/D. 3) Signal duplexing for two-wire transmission. 4) Telephone line interface. Most manufacturers provide multiple part solutions that comprise a Class-2 fax subsystem; some hybrid component manufacturers provide single part fax modem solutions. Although more expensive, the single part solution best meets the design assumptions made here. The host hardware interface is well defined (RS-232), the telephone network interface is already FCC type accepted, everything in between is "tested," and PCB space is minimized (e.g., 1.2” x 1.8” ).
A Summary Examination of: EIA/TIA 592, CCITT T.4 and T.30.

Given that we have identified the use of Class-2 fax modem technology, we will now examine what related standards participate in the sending of a fax. The ISO/OSI 7-layer model is deployed to describe the layered relationships between communication protocols. Figure 7 shows a “protocol stack” that provides an approximation of the interrelationship between CCITT and EIA/TIA standards that are used to send a fax via a Class-2 device. The protocol and accompanying standard at each level is in itself a subject for detailed discussion, therefore only a synopsis of what each level attempts to accomplish is given.
The protocol stack shown (figure 7) is not "perfect" in that the interface between any two levels is not clean and crisp. The host application running on the existing embedded system may be responsible for intermediate work in almost all of the layers, (excluding the physical layer) during fax transmission. The work that must be performed at any one level is usually minimal, except for T.4 at the presentation layer and flow control during interrupt service.

When a call is placed to send a fax, the host application sends "AT" commands to the Class-2 fax modem to initialize the fax subsystem, set default session parameters, and establish a connection (dial) to a remote fax machine. Intermediate session "AT" commands may be issued between pages and finally to release facilities (hang-up). Class-2 fax modems essentially implement a super-set of standard modem "AT" commands found in most 2400 bps modems. There are well over 50+ additional "AT" commands for controlling the Class-2 fax subsystem and their general syntax is "AT+F<tokens...>".

An example session (see figure 13) to send two fax pages might include the following steps:

1. Host sends initialization commands are sent to configure the Class-2 fax modem.
2. Host provides optional identification and capability data.
3. Host dials number, call is placed; host waits for "CONNECT."
4. Local and remote fax subsystems negotiate modem speed, and exchange capability information.
5. Host receives remote fax device capabilities and identification in a T.30 DIS frame.
6. Host encodes scan line information (T.4) and sends it while performing some aspects of flow control and timing management.
7. Host sends end of page mark and waits for T.30 "MCF" and/or "CONNECT."
8. Host issues hang-up commands.

The bulk of the work in managing the data link and framing the information is handled by the T.30 protocol and it is implemented by the Class-2 fax subsystem. Therefore the host embedded system has three major responsibilities:

1. Issuing "AT" commands and appropriately responding to them.
2. Translating scanlines to Huffman code runs for transmission.
3. Providing device driver and interrupt service routines.

A Class-1 fax subsystem places more of the T.30 session management responsibilities on the host application and this reaffirms our selection of a Class-2 fax subsystem. Given that we have described the basic semantic interactions with a Class-2 fax modem, a candidate configuration is now selected and its impact on hardware and firmware design is discussed.

**CANDIDATE HARDWARE/FIRMWARE CONFIGURATION**

The interface to Class-2 modem devices is either bus/port parallel, or serial (e.g. RS-232). The hybrid single part subsystems typically use only an RS-232 serial interface. If an
existing embedded platform has an available serial interface, as is the case with popular embedded controllers such as the 68HC11, 68HC16, 8051, and 8018X derivatives, it can be utilized. However, if a serial interface is not available, an additional UART may be needed as is shown in figure «HostSystemFigure». The addition of this UART may effect the economies of the design for larger production quantities. The selection, however, of the extravagant 16550 UART may be well justified, as it provides a 16 byte FIFO that reduces the bursty interrupt traffic on the serial link and frees valuable processor cycles.

![Host System Diagram]

**Figure 8**

The integration of the Class-2 fax subsystem places the following firmware responsibilities on the embedded system designer:

1. **Host Application(s)**

   1. Issue and respond to “AT” and “AT+F” commands.
   2. Respond to T.30 session status; most status messages can be ignored.
   3. Create a fax page bit map either in physical or virtual form.
   4. Translate a scanline bit map to T.4 Huffman code runs.
   5. Adapt to the remote device’s printing capabilities (timing) and manage flow control.
   6. Interact with the existing application to determine when a fax should be sent.
   7. Manage a list of phone numbers, and track when and to whom a fax is sent.

2. **The FaxAPI**

   The “FaxAPI” is a layer of software that further hides T.4, Class-2 and T.30 details and allows the application developer to treat the fax subsystem as an abstract entity. An ANSI C code example in figure 9 shows how simple it can be to send a text message to a fax machine. Implementing a FaxAPI layer has the advantages of allowing the physical and logical interface to the fax subsystem to change without affecting existing code. This layer
allows the user to specify ASCII text, primitive objects (e.g., circles, lines, fill patterns, etc.), and complex objects (e.g., gauges, meters, bar graphs, line graphs, and bit maps) for transmission to the remote fax machine. These objects are then translated into Abstract Data Type (ADT) forms in preparation for T.4 translation.

3. T.4 Translation

This level is responsible for performing the translation of ADT representations generated by the FaxAPI layer to Huffman codes as prescribed by the CCITT T.4 standard.

4. UART Device Driver

1. The device driver and ISR9 must accommodate the time sensitive transmission of bursty asynchronous data blocks that range from 8 to 1K in size with a data rate of 19.2 kbps.
2. Provide XON/XOFF and/or RTS/CTS flow control.
3. Adapt to the remote device's timing and printing capabilities.

Recall that the G3 fax will send data in synchronous fashion at 9.6 kbps. Data is presented to the Class-2 fax subsystem at a rate of 19.2 kbps in asynchronous form and this accounts for its bursty nature. The fax subsystem DSP maintains an elastic buffer that will always have enough bytes in it to maintain the integrity of an unbroken 9.6 kbps synchronous bit stream. This is actually quite useful in that it gives the embedded system some time to do other things between bursts. It will be the responsibility of the host device driver to handle the flow control (e.g., XON/XOFF) and timing constraints outlined in figure 4.

### Code Snippet

```c
extern int FaxSendReport( void )
{
    auto FAXSessionHandle hFaxSession;
    auto char FaxMessageBuffer[ 48 ];

    if( ( hFaxSession == FAXOpenSession("5552368") ) ! NULL )
    {
        sprintf( FaxMessageBuffer,
            "Station: %03d; Temp: %03d; Pressure: %03d\n",
            GetStationId(), GetTemp(), GetPressure() );
        FAXWrite( hFaxSession, FaxMessageBuffer );
        FAXCloseSession( hFaxSession );
        return( PASS );
    }
    return( FAIL );
}
```

**Figure 9**
The one question the embedded system designer now asks is “How much memory will all this cost?” The answer is based upon the processor, compiler technology (if implemented in ANSI C) and what type of information needs to be transferred. A minimal system that can only send text based faxes has been implemented in ANSI C for less than 24K ROM/8K RAM using the 80C517. These numbers will, of course, vary between C compilers, instruction sets, and hardware configurations. A richer implementation that is capable of sending both text, graphics, and bit map images will consume more memory. The FaxAPI layer can be made less general or done away with all together if ROM space is scarce. This miser’s philosophy does dramatically affect the extendibility and maintenance of your embedded fax software in the future.

Several companies and organizations have formulated fax APIs. Unfortunately their designs and implementations require most, if not all of, the facilities listed in figure 1 and are therefore unusable in the embedded system environment.

**DESIGN ISSUES: PROCESSOR OCCUPANCY AND SCHEDULING**

A principal issue facing the embedded system designer desiring to integrate fax capability is the architecture of the existing software. Many embedded system software architectures are at best “ad-hoc” in that there is rarely a formal preemptive kernel (executive) or uniform system services. Tasks in this type of environment are cooperative and voluntarily relinquish control to one another — there is no formal scheduling mandated from a central entity. This is an intricate environment in which to work — a fax subsystem has time sensitive requirements and must be “woven” into an existing cooperative task hierarchy.

Embedded system architectures that use a layered preemptive kernel that provides resource management, synchronization, and scheduling makes it much easier to adapt a Class-2 fax subsystem. Synchronization (e.g., messages, semaphores) between the user’s task and the fax subsystem is formal; necessary time slice requirements and priorities can be requested of the kernel, and the existing task architecture need not be directly affected.

In both cooperative and preemptive scheduling scenarios, sending a fax does not really become a burden on the processor until the sending modem starts streaming encoded scanline data at 9.6 kbps. When this occurs you can expect about 2000 interrupts/second with 1 byte UART FIFO or 125 interrupts/second with a 16 byte UART FIFO in bursts of about 240 ms in length. There are about 200 ms of “quiet time” between bursts for the processing of other system tasks or fax information (e.g., T.4 encoding). This type of interrupt traffic can challenge, or even corrupt, the integrity of an embedded system that must maintain realtime integrity by making heavy processor occupancy demands. Few systems have the luxury of abandoning or jeopardizing their primary mission in order to send a fax.

The designer must therefore assess current scheduling practices, determine the amount of “vacant” usable processor occupancy, evaluate interrupt latency and overhead, and memory
requirements. All of which are a function of the processor instruction set, current tasks and their priorities, type of UART used, and the implementation of a FaxAPI and device driver.

ISSUES IN SENDING BLOCK DATA (CLASS-2)

All Class-2 (and SendFax) devices listed in figure «FaxModemVendorsFigure» are capable if 2.4 kbps FDX (full duplex) transmission. Some supply MNP5 and V.42bis correction and compression. Note that when operating in FDX data mode, the fax subsystem behaves as a regular “AT” driven data modem. If, however, a fax subsystem has the ability to send and receive a fax using Class-2 technology, it is possible to send HDX (half duplex) data at 4 (9.6 kbps) or even 6 (14.4 kbps) times the 2.4 kbps FDX rate. There are, of course, some limitations:

1. Although T.30 does provide methods by which errors can be detected and corrected, its framing was designed on the assumption that bit errors could be tolerated. Correcting errors using T.30 is inefficient.
2. Various unique bit patterns will cause the T.30 state machine at both ends to return to control mode; DLE (Data Link Escape) processing, byte insertion, is required.
3. T.30 could train and establish the data link speed at 4800 bps HDX which may yield an effective information rate below that of V.42bis at a 2400 bps data rate.
4. A CCITT T.434 Binary File Transfer (BFT) standard has been proposed and can be used to transmit large blocks of data, but support for BFT is not yet integral in many of today’s Class-2 modem products.
5. An additional link layer for framing and packet identification may have to be added so full and partial block retransmission can be supported.

Using today’s existing T.30 protocol, a data block is essentially a fax page. When the block (page) is sent, the remote fax subsystem can acknowledge its proper or improper reception. If an error has occurred, the calling fax subsystem can re-transmit the block. When compared to other HDX HDLC\textsuperscript{10} protocols T.30 is inefficient. It is therefore important that an optimal message block size \( M \) be found to get reasonable block transfer performance using the fax subsystem. \( M \) is determined as a function of several parameters such as noise on the link, path turnaround time, etc. Dixon Doll presents a closed form equation for estimating the optimal length of a block. For an average telephone connection using timing from a “typical” Class-2 fax subsystem \( M \) can be computed using the equation below and ranges from 4 to 16 kbytes result.

\[
M^* = \frac{C - RT}{2} + \sqrt{\frac{R^2T^2}{4} + \frac{CRT}{2} + \frac{C^2}{4} + \frac{RT+C}{2(1-K_2K_3E)}}
\]
There are various forms of FEC (Forward Error Correction) that can be employed so that if errors do occur during transmission, the receiver can make necessary corrections without a retransmission. The cost of FEC in firmware (ROM space) is too high for typical embedded systems and the cost of FEC hardware is more than current V.32 FDX (9.6 kbps) modem subsystems with V.42bis data compression.

Today’s Class-2 fax subsystem can be called upon to efficiently transmit large amounts of data that can tolerate errors. Such data would typically have enough redundancy so that lost or corrupt data can be “reconstructed” through simple interpolation or extrapolation. Such types of data might include uncompressed video (bitmap) or a sequence of pressure readings. When CCITT T.434 BFT (Binary File Transfer) protocol becomes ubiquitous in Class-2 modems then even more efficient options for transferring large blocks of data become viable.

### FAXAPI AT COMMAND SET EXTENSION

The pervasive Class-2 AT command set can be augmented with commands that allow the applications programmer to interact with the fax sub-system as if it were just a modem device. Figure 10 shows an example session that lists some additional AT commands that have been added in one implementation of a fax sub-system. This technique is especially useful if the fax sub-system has a dedicated microcontroller with two serial ports. One port is dedicated to the fax modem and the other interfaces to a client microcontroller program issuing only “AT” commands. The client microcontroller is then freed from all T.4 and device interrupt processing responsibilities — a significant burden on some existing 8-bit platforms.
AT++FAXAPI line 100 100 200 200 1
OK
AT++FAXAPI text 20 20 "This is the first line of text" 1
OK
AT++FAXAPI text 50 20 "This is the second line of text" 1
OK
AT++FAXAPI box 10 10 60 1000 1
OK
AT++FAXAPI send "5552368" CSI "Test FAX"
CONNECT 9600
...
CFR

Figure 10

On an existing microcontroller system this is possible with the addition of a very simple lexical analyzer and parser to intercept the AT++FAXAPI commands and essentially pass all other AT commands on to the fax sub-system. Should an additional microcontroller be summoned to process these additional features the task becomes more complex as another hardware and software effort must be undertaken. This may not be a choice should the existing microcontroller platform not have enough processor occupancy or memory space available. Good candidate microcontrollers for this task are the Siemens SAB 80C168, the Zilog Z-180, and the Intel 80C188EB/EC.

FUTURE EXPECTATIONS

EIA/TIA 592 (Class-2) has been recently approved (/YY). Full Class-2 capabilities are still evolving at the fax subsystem level as manufacturers strive to implement EIA/TIA 592. Expect manufacturers of single part fax subsystems to provide a full Class-2 implementation within the next eighteen months.

The ability to receive fax data is a recent addition to many Class-2 platforms and this opens the door to many interesting possibilities. One exciting possibility is for the embedded controller to receive information from a fax machine that may contain information coded much like a "bubble-in" score sheet. With the proper authentication, a remote embedded system can be commanded or re-configured from any fax terminal. Xerox has demonstrated this concept by using its "Glyph" encoding technology in its PaperWorks product. Xerox has hinted that it may open up this technology by licensing it to OEMs.

The recent EIA/TIA 605 packet protocol is another standard of interest. This protocol insures that as a fax is received, data from the fax subsystem (DCE\textsuperscript{11}) is packetized with a check sequence. This allows the embedded system (DTE\textsuperscript{12}) to determine if bytes were lost due to a high interrupt latency at the UART and request a re-transmission from the local fax subsystem (DCE). Embedded systems designers that must meet stringent real time...
constraints should strongly consider fax subsystems that implement EIA/TIA 605. Although a horribly inefficient way to store "information," fax mail boxes could be used to store, forward and re-route fax transmissions from embedded systems. This, for example, would allow field personnel to pick-up status reports originating from embedded platforms at any fax machine at their convenience. Of course, a more efficient way to transfer information would be to dial-up a central server from a notebook computer and transfer information in much more efficient binary forms. Using a fax mailbox is a useful option since almost all offices, airports and hotels have faxes that can be used as terminals. The fax mailbox server can be provided by the phone company, 3rd party service bureau, or a dedicated PC.

There are several organizations and alliances that have been formed to draft and promulgate fax standards. Unfortunately none really address issues important for embedded systems. These organizations are, however, instrumental in providing a vehicle by which issues can be raised and addressed. In addition to the CCITT study groups and EIA/TIA standards committees are the ICFA (International Computer Fax Association) and the FaxBIOS Association.

SUMMARY

With the proper assessment of an embedded system's resources (hardware and firmware) a Class-2 fax subsystem can be added to provide remote printing capabilities on the most popular printer in the world. Although most of today's Class-2 fax subsystems support V.42bis compression at a data rate of 2400 bps, the fax subsystem can be called up to efficiently send large amounts of data that can tolerate some errors. Activity in the standards arena and forthcoming products from large and small companies hold promise for even more advanced fax capabilities in the future.

The plethora of PC based fax products (hardware and software) ranging from Windows applications to complete fax servers essentially makes the fax the most universal medium in which to transmit, store, and exchange graphic and text information — the embedded system designer now has the power to take advantage of all that.
APPENDIX

DESCRIPTION OF THE ISO PROTOCOL STACK

<table>
<thead>
<tr>
<th>Standard</th>
<th>In</th>
<th>Out</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EIA/TIA 592</td>
<td>AT commands</td>
<td>AT responses T.30 responses</td>
<td>“AT” commands are issued and their semantics are executed. A response is returned (e.g., OK or ERROR). Used to interact with T.30, predominantly in Phases A, B, D and E. AT commands can elicit responses from the T.30 session layer.</td>
</tr>
<tr>
<td>CCITT T.4</td>
<td>Scan line bit map</td>
<td>Huffman code runs</td>
<td>provides translation between a one dimension bit map to Huffman codes; frame lines, and pages; provide one and two dimensional coding; provide optional resolution for detected errors.</td>
</tr>
<tr>
<td>CCITT T.30</td>
<td>Concatenated Huffman code runs Instructions from EIA/TIA 592</td>
<td>Link and session management</td>
<td>Manages the basic phases of a fax transmission. (A) call setup, (B) pre-message procedures, (C) message transmission, (D) post-message procedure, (E) call release.</td>
</tr>
<tr>
<td>HDLC</td>
<td>Data stream</td>
<td>framing, timing, error detection</td>
<td>provide synchronous framing and error detection capabilities.</td>
</tr>
<tr>
<td>CCITT V.29</td>
<td>digital bit stream</td>
<td>analog QAM</td>
<td>convert digital pulses to an analog signal that varies in phase and amplitude.</td>
</tr>
</tbody>
</table>

Figure 11

CCITT T.30 PHASES OF FAX TRANSMISSION

<table>
<thead>
<tr>
<th>Φ</th>
<th>T.30 Semantics</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Call establishment</td>
<td>Ring, answer, determine if FAX or data. CS1 passed as V.21</td>
</tr>
</tbody>
</table>
| B | Negotiate capabilities | Exchange capabilities; e.g., data rate, scanning speeds, dimensions, 1 or 2 D encoding, etc.  
Train modems; try [14.4,] 9.6, 7.2, 4.8, 2.4, 0.3 kbit data rates. Try highest speed and fall back if there is a failure to train.  
Adaptive equalization, adjust for group delay distortion, phase jitter, gain loss, etc.  
Determine the remote machine's capabilities. |
| C | Message Transmission | Send scan lines; delimiters are EOL, and 6*EOL.                                |
| D | Post Message (RTC) | After receipt of 6*EOL; possible partial page; possible next page.                |
| E | Call release       | Drop local loop to (class 5) end office.                                        |

Figure 12
## EIA/TIA 592 EXAMPLE SESSION

<table>
<thead>
<tr>
<th>DTE Commands</th>
<th>DCE Responses</th>
<th>Local Action</th>
<th>Remote Action</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>AT+FCLASS=2.0</td>
<td>OK</td>
<td>Set mode to EIA/TIA 592</td>
<td></td>
<td>DTE may load a local ID</td>
</tr>
<tr>
<td>AT+FLI=&quot;FaxId&quot;</td>
<td>OK</td>
<td>Set local ID</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ATDT5552368</td>
<td>+FCO [+FNF: &quot;&lt;nsf&gt;&quot; ]</td>
<td>off hook, dial, send CNG detect flags [get NSF] [get CSI] get DIS</td>
<td>answer, send [CED], Preamble, [NSF], [CSI], DIS</td>
<td>DTE may respond to NSF frame if +FLI loaded</td>
</tr>
<tr>
<td>AT+FNS=&lt;hex NSS FIF string&gt;</td>
<td>OK</td>
<td>[send NSS,] [send TSI,] send DCS send TCF get CFR send carrier</td>
<td>[get NSS,] [get TSI,] get DCS get TCF send CFR receive carrier</td>
<td>if +FNS loaded</td>
</tr>
<tr>
<td>AT+FDT</td>
<td>{+FCS:&lt;codes&gt;} CONNECT</td>
<td>send page 1 T.4 data [send RTC,] send MPS get MCF</td>
<td>get 1 page T.4 data get RTC get MPS send MCF</td>
<td></td>
</tr>
<tr>
<td>AT+FDT</td>
<td>CONNECT</td>
<td>send carrier</td>
<td>receive carrier</td>
<td></td>
</tr>
</tbody>
</table>

**Figure 13**

### MAINSTREAM FAX MODEM MANUFACTURER

<table>
<thead>
<tr>
<th>Manufacturer</th>
<th>Family</th>
<th>Send Fax, EIA 592 Capabilities beyond CCITT G3/T30; EIA 578 Class 1, Advanced echo cancellation.</th>
</tr>
</thead>
<tbody>
<tr>
<td>AT&amp;T</td>
<td>WE DSP16A</td>
<td></td>
</tr>
<tr>
<td>Cermetek</td>
<td>CH1782</td>
<td>✓ Integral DAA; V.21: one package.</td>
</tr>
<tr>
<td></td>
<td>CH1785</td>
<td>✓ Integral DAA; V.21: one package.</td>
</tr>
<tr>
<td></td>
<td>CH1792/1793</td>
<td>✓ Integral DAA; V.21: one package.</td>
</tr>
<tr>
<td>Cirrus</td>
<td>MD9624AT/EC</td>
<td>Class 1: Voice Codec.</td>
</tr>
<tr>
<td>Exar</td>
<td>XR-2900+ XR2943</td>
<td>V.42; MNP-5.</td>
</tr>
<tr>
<td>Phylo</td>
<td>9202 9105</td>
<td>✓ ADPCM Voice/CODEC; V.29; V.17; Caller ID</td>
</tr>
<tr>
<td>Rockwell</td>
<td>RC96V24</td>
<td>✓ V.42 bis; MNP-10: ADPCM Voice/CODEC V.32bis V.42bis MNP-10: ADPCM Voice/CODEC V.42bis.</td>
</tr>
<tr>
<td></td>
<td>RC144ACL</td>
<td>✓ V.42bis.</td>
</tr>
<tr>
<td>Sierra Semiconductor</td>
<td>SC11064+</td>
<td>✓ V.22.</td>
</tr>
<tr>
<td></td>
<td>SC11643</td>
<td>✓ V.22: Caller ID;</td>
</tr>
<tr>
<td></td>
<td>SC11054</td>
<td>✓ V.22.</td>
</tr>
<tr>
<td></td>
<td>SC11055</td>
<td>✓ V.22.</td>
</tr>
<tr>
<td>Silicon Sytems</td>
<td>SS1 732D417</td>
<td>✓ V.22bis; MNP-5 (preliminary data)</td>
</tr>
<tr>
<td>Xecore</td>
<td>XE2946M</td>
<td>✓ ✓ ✓ Integral DAA; V.21: one package.</td>
</tr>
<tr>
<td></td>
<td>XE9624</td>
<td>✓ ✓ ✓ Integral DAA; V.23: V.42; MNP-10.</td>
</tr>
<tr>
<td></td>
<td>XE9696V</td>
<td>✓ ✓ ✓ Integral DAA; V.32bis: V.42bis; MNP-10: ADPCM Voice/CODEC</td>
</tr>
<tr>
<td>Yamaha</td>
<td>YTM40B+ GTM407</td>
<td>✓ ADPCM Voice/CODEC; Caller ID; DAA interface; microcontroller firmware</td>
</tr>
</tbody>
</table>

**Figure 14**
CCITT V SERIES MODEM STANDARDS

<table>
<thead>
<tr>
<th>CCITT Standard</th>
<th>Fax/Data</th>
<th>Bps</th>
<th>Baud</th>
<th>Modulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>V.17</td>
<td>F</td>
<td>14000</td>
<td>2400</td>
<td>TCM</td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>12000</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>9600</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>7200</td>
<td></td>
<td></td>
</tr>
<tr>
<td>V.21</td>
<td>D</td>
<td>300</td>
<td>300</td>
<td>FSK</td>
</tr>
<tr>
<td>V.21 Channel 2</td>
<td>F</td>
<td>300</td>
<td>300</td>
<td>FSK</td>
</tr>
<tr>
<td>V.22</td>
<td>D</td>
<td>1200</td>
<td>600</td>
<td>DPSK</td>
</tr>
<tr>
<td>V.22bis</td>
<td>D</td>
<td>2400</td>
<td>600</td>
<td>DPSK</td>
</tr>
<tr>
<td>V.23</td>
<td>D</td>
<td>1200 forward channel</td>
<td>300</td>
<td>QAM</td>
</tr>
<tr>
<td></td>
<td>D</td>
<td>75 back channel</td>
<td>75</td>
<td></td>
</tr>
<tr>
<td>V.26</td>
<td>D</td>
<td>2400</td>
<td>1200</td>
<td>DPSK</td>
</tr>
<tr>
<td>V.27ter</td>
<td>F</td>
<td>4800</td>
<td>1600</td>
<td>DPSK</td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>2400</td>
<td>1200</td>
<td>DPSK</td>
</tr>
<tr>
<td>V.29</td>
<td>F</td>
<td>9600</td>
<td>2400</td>
<td>QAM</td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>4800</td>
<td>2400</td>
<td>QAM</td>
</tr>
<tr>
<td>V.32</td>
<td>D</td>
<td>9600</td>
<td>2400</td>
<td>TCM</td>
</tr>
<tr>
<td></td>
<td>D</td>
<td>4800</td>
<td>2400</td>
<td>TCM</td>
</tr>
<tr>
<td>V.32bis</td>
<td>D</td>
<td>14400</td>
<td>2400</td>
<td>TCM</td>
</tr>
<tr>
<td></td>
<td>D</td>
<td>12000</td>
<td>2400</td>
<td>TCM</td>
</tr>
<tr>
<td></td>
<td>D</td>
<td>9600</td>
<td>2400</td>
<td>TCM</td>
</tr>
<tr>
<td></td>
<td>D</td>
<td>7200</td>
<td>2400</td>
<td>TCM</td>
</tr>
</tbody>
</table>

Figure 15

BIBLIOGRAPHY & REFERENCES


“Data Communications Over the Telephone Network,” The International Telephone and Telegraph Consultative Committee; CCITT Blue Book; Series V Recommendations; Recommendations V.1 - V.230; Volume VIII - Fascicle VIII.1; IXth Plenary Assembly; Melbourne, 15-25 November 1988.

“Data Communication Networks, Transmission, Signaling and Switching, Network Aspects, Maintenance and Administrative Arrangements,” The International Telephone and Telegraph Consultative Committee; CCITT Blue Book; Series X Recommendations; Recommendations X.40 - X.181; Volume VIII - Fascicle VIII.3; IXth Plenary Assembly; Melbourne, 15-25 November 1988.


"RC96ACL Low Power Integrated Data/Fax/Voice Modem Device Set (29200N86)," 1992 Data Sheet; Rockwell Corporation, 4311 Jamboree Road, Newport Beach, CA 92660-3095; 714.833.4655 (Tel.) 714.833.6898 (Fax.).

"Standard Products," 1992 Data book; Sierra Semiconductor Corporation, 2075 North Capitol Avenue, San Jose, CA 95132. 408.263.9300 (Tel.), 408.263.3337 (Fax.).

"Databook on Telecommunication Products," 1992 Data book; Exar Corporation, 2222 Quem Drive, San Jose, CA 95161; 408.434.6400 (Tel.), 408.943.8245 (Fax.).

"Integrated Circuits for Communication Products," 1991 Data book; Silicon Systems, Inc., 14351 Myford Road, Tustin, CA 92680; 714.731.7110 (Tel), 714.669.8814 (Fax).

1 There are of course products available for the Macintosh as well, they too suffer the same consequences listed for PC based products.
2 Several vendors do offer a standard platform across several instruction sets; but the interface between any two vendors seldom aligns. Realtime POSIX addresses this dilemma, however, there is currently no usable standard.
3 International Telegraph and Telephone Consultative Committee
4 Electronic Industries Association/Telecommunications Industry Association
6 International Standards Organization/Open System Interconnection
7 Universal Asynchronous Receiver Transmitter
8 Fax Application Programming Interface
9 Interrupt Service Routine (typically integral with device driver design).
10 High level Data Link Control
11 Data Communications Equipment (fax modem) (a.k.a. DCTE: Data Circuit Terminating Equipment)
12 Data Terminal Equipment (embedded system)
13 This table does not include all current and/or future offerings from all manufacturers. Products and highlighted features were selected on the basis of their applicability in an embedded system.
14 SendFax® is a registered trademark of Sierra Semiconductor
15 EIA/TIA 592 (Class-2) is such a recent standard that no known manufacturer complies 100%. A check mark in this column indicates that the manufacturer implements a proper subset of EIA/TIA 592.