

DESIGN AND IMPLEMENTATION OF A GENERIC TESTBED FOR ADAPTING ATM FOR THE TACTICAL RADIO ENVIRONMENT

Phil Webb, Dave Lindsay, Dr Paul Thorlby, Peter Loretto (on secondment from BAe) & Scott Eagan (on secondment from Australian Military)

Communication Department, Command & Information Systems Sector, Defence Evaluation & Research Agency, Malvern
Worcestershire, WR14 3PS, UK.

Abstract-Asynchronous Transfer Mode (ATM) technology offers many technical advantages to the military user and is of a great interest to the military for their future army communication requirements. One of the many challenges being faced when using ATM in the military arena is improving its robustness and performance at bit error rates (BER) of 10^{-5} and higher. These BER rates are inherent to military wireless links.

Many “link adaptation” schemes have been modeled with various error profiles; these have given invaluable error performance figures showing how ATM may survive on an error prone link. Our interest lay in designing a generic testbed of flexible architectures to enable us to try out various schemes in real environments with minimal engineering overheads and gain real results.

This paper discusses the thinking behind the design methodology, the architectures used for implementation and the final “scheme” chosen for proof of concept testing.

Introduction

The main philosophy behind the use of Asynchronous Transfer Mode (ATM) for military communication is to take advantage of the considerable research and investment the commercial world has carried out on ATM [1]. This advantage would cut the cost considerably of any future military communication system employing this technology. This cost reduction is at its maximum if equipment is procured without modification, the slightest modification to any commercial equipment, be it hardware or software, would remove all advantages originally gained from the use of commercial protocols. To keep this advantage it is important that the ATM protocol is only altered when required (i.e. before a radio shot) and then all alteration to the ATM cell removed before the network sees it again. This makes the link hardener transparent to the network and the full commercial advantage is gained.

The New ATM Cell Structure

The adapted ATM cell should have an increased resilience to errors and have a robust synchronization scheme suitable for

the wireless environment. These requirements obviously give a wide variety of options on how we can adapt the cell for the tactical radio environment. This paper describes the development of a flexible modular testbed architecture upon which prototypes of ATM link hardening schemes could be rapidly developed.

To reduce the complexity of the initial “proof of concept” implementation, a scheme with minimal bandwidth overhead was chosen and is described in this paper to show how the testbed can be utilised. This paper does not cover the advantages and disadvantages of differing schemes, further research is being carried out on this and will be described in a future paper.

The chosen scheme inserts radio sync to the ATM cell and adds a Bose Chaudhuri and Hocquenghem (BCH) block code to the ATM header.

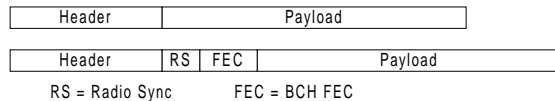


Figure 1 – Standard ATM Cell and the Adapted Cell

Forward Error Correction Scheme

In order to minimize the effect of the tactical environment we employ a BCH block code on the cell Header. The code that has been chosen is 63/45 block code that can correct up to 3 errors in the 63 bits, this enables the header to survive in error rates of around $10^{-3} / 10^{-2}$. The ATM cell header is 40 bits long (5 octets) this gives us 5 spare bits in which we can insert the radio sync after the cell has been encoded. This gives ~5% overhead on the channel. In this particular scheme the payload is left unaffected.

Radio Synchronization Scheme

The standard method of gaining sync in ATM requires the cell boundaries to be known. Unfortunately this information is

internal to the hardening process and can only be used after the decoding of the Forward Error Correction (FEC). This means we are required to provide radio synchronization outside of this hardening process to enable sync to be gained at the far end. To do this we use a 15 bit Pseudo Random Bit Sequence (PRBS) generated from a 4 bit generator polynomial, this is split across 3 cells so that 5 bits are in each cell in the space provided by the BCH block code as described earlier. The received PRBS is then correlated at the receiver with a stored version to determine the location of the cell boundary.

Design Approach

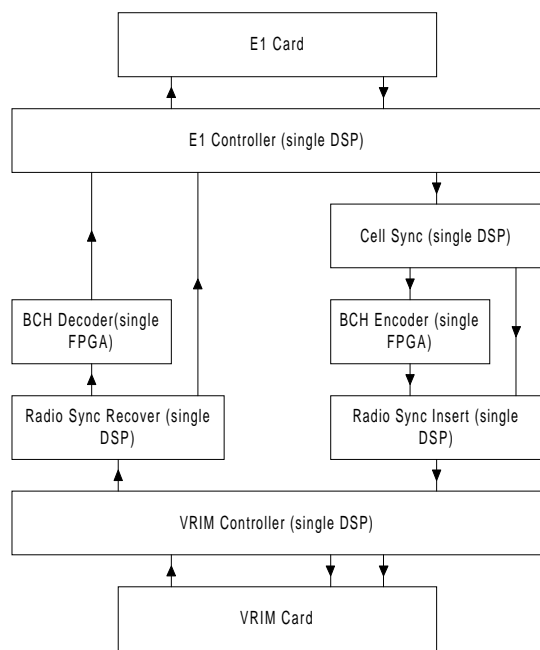


Figure 2 – Hardware layout showing modular code design

The Link Hardening testbed is built on a VME back plane connected to a SUN workstation. Using a number of DBV44 Digital Signal Processor (DSP) carrier boards (connected directly to the VME back plane) each one supporting up to 4 single TMS320C40 DSP TIM modules or Field Programmable Gate Array (FPGA) TIM modules. With this hardware we have developed a very flexible and modular design environment. Each DSP TIM module has six communication ports, some of these can be patch paneled on the front of the DBV44 card to any other DSP and some are hardwired by the DBV44 internally. Data can be passed between devices by Direct Memory Access (DMA) transfer through any of the six 32 bit communication ports on the modules.

The FPGA modules contain a Xilinx 4025 Synchronous Random Access Memory (SRAM) based FPGA and two 4010 FPGA's, also SRAM based. The two 4010's act as the six

communication ports so that the modules look like C40 DSP's to the other modules, this enables us to DMA data to and from the FPGA. The 4025 is a large FPGA with 25000 logic gates. Due to the device being SRAM based it is very efficient at implementing RAM, this means that it can store data before it carries out any processing and makes it very powerful for large data crunching (ideal for cell by cell processing). This module has been developed under contract by TRL in England.

The Link Hardening Testbed interfaces to a 2 Mbit/s E1 port on any ATM switch through an E1 card which handles all the E1 framing issues and presents 15 bit word format data to the processor.

On the radio side we have a Variable Rate Interface Module (VRIM) card this has been developed under contract by System Engineering and Assessment (SEA), England. This can interface to a variety of military radios from 300 baud to 2Mbits/s and has a standard C40 DSP 32 bit comm port interface.

Modular Design Approach

As can be seen in figure 2, the hardware has been configured in such a way as to aid with the modular design philosophy. As mentioned earlier the communications between devices are via the DSP's DMA coprocessor. This speeds up data transfer and enables us to split the code into very distinct sections, each section is mapped onto the individual DSP sites shown in figure 2, and gives us the flexibility in our testbed to increase our processor or FPGA count easily for future more complicated schemes.

Software Design

The software is separated into sections so that it can be mapped easily onto the separate hardware devices shown in figure 2. Data flow through this modular software design is shown below in figure 3.

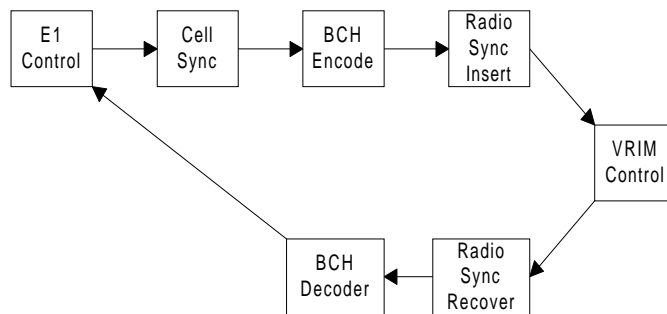


Figure 3 – Software data flow through the modules

The software is split so that each major function is downloaded onto a single device, this gives us the modular flexibility required to be able to increase the functionality of the testbed easily.

Transmit Side Software functionality

1) *E1 Controller*: The E1 Controller DSP controls the E1 card. The E1 card handles all the E1 framing and hands the DSP raw unsynchronised ATM. The whole process is interrupt driven and very timing critical. The E1 card produces an interrupt every 125µs (a single E1 Frame) and passes 240 bits, 30 channels worth of data in 15bit words to the DSP. The DSP has two tasks, the first is to buffer the E1 data and wait until it has two ATM cell's worth of data and then it passes it to the cell sync DSP which carries out cell delineation. The second is on the receive side, this passes data to the E1 card in 15 bit word format when the E1 card requests it. If a request comes through from the E1 card and the DSP does not have enough data to transmit then the DSP must insert an idle cell into the data stream to keep the 2Mbit interface rate with the switch (this would occur if the radio links bandwidth were less than 2Mbits/s).

2) *Cell Sync*: The cell delineation routine uses the cell Header Error Check (HEC) byte to enable it to find the cell boundary just as a switch would. The cell sync DSP receives data from the E1 controller card. On each DMA interrupt the DSP receives two cells worth of data. The cell sync state machine is shown below:

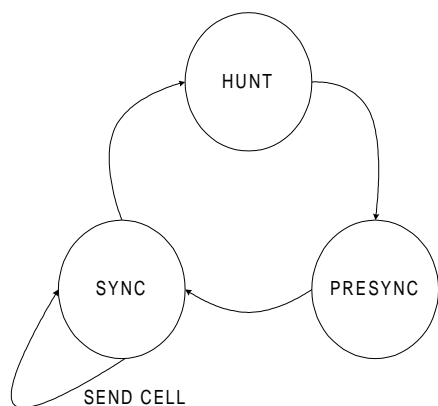


Figure 4 – Cell Delineation state machine

The routine stays in the HUNT state until it has found a valid header and cell boundary. It then moves on to PRESYNC. In this state the routine checks to see if what it found in the HUNT state was actually a valid header, it does this by checking for 4

more headers in the assumed boundary positions. When this has been achieved the routine moves into the SYNC state. In this state the code is continuously transmitting valid cell headers to the FPGA and cell payloads to the Insert Radio sync DSP via DMA. As the routine finds valid cell boundaries the code checks to see if they are idle cells. When it finds an idle cell it removes it, this means that only data cells are encoded, this then enables the bandwidth to be matched correctly at the radio interface.

3) *BCH Code*: The BCH coding and decoding algorithms have been designed in hardware using VHDL and implemented on reconfigurable logic, this gives a very large increase in speed over the software alternative. The FPGA interface has been designed to the Texas Instrument TIM site specification by TRL, in England. The 4025 is presented with a 32 bit word and a strobe for timing. These devices are running at 50 MHz so the encoding delay is approximately 70 clock cycles (140ns). The FPGA receives the two 32 bit words from the cell sync routine, 40 bits are header all the rest are set to zero. The 4025 then encodes the data using the 63/45 BCH code. The standard ATM cell header is 40 bits long therefore this BCH code will leave an extra 5 bits, this is for the radio sync to be inserted into. These 5 bits are left blank by the 4025 FPGA. The encoded header is 63 bits long, but is passed from DSP to FPGA and back again as two 32 bit words. This leaves an extra bit at the end of the encoded header. This “stuff bit” is not removed when the header and the payload are reassembled, it is simply taken as an allowable overhead.

4) *Radio Sync*: Radio synchronisation works on a 15 bit PRBS sequence, this is added to the header before the cell is transmitted to the VRIM card. Figure 4 shows the structure of the radio frame.



Figure 5 – Radio Frame Structure

The 15 bit sequence is split into three 5 bit words each word is then inserted into the space provided by the FEC scheme. After the sync has been inserted the cells are passed on to the VRIM controller DSP.

5) *Radio Interface*: The VRIM handles bandwidth matching on the radio side. The radio interface can be configured to a variety of bit rates from 300 baud to 2 Mbits/s. The VRIM card is configured on download of software and can not be

dynamically reconfigured. The VRIM card requests data from the VRIM controller card when its buffers reach a minimum point. If the input data rate from the switch is less than the radios available bandwidth then the VRIM controller DSP will insert fully encoded idle radio frames whenever a request from the VRIM occurs and the Controller has no data. If the data rate on the input should be greater than the available radio bandwidth then cell loss will occur due to buffer overflow in the transmit system.

Receive Side

On the receive side the radio frames are buffered by the VRIM card and converted to 32 bit word format for the DSP environment.

1) Radio Sync Recovery: The 32 bit words are passed straight through to the radio sync recovery DSP. The DSP will buffer the data until it receives a whole radio frame of data. Once the whole radio frame has been received the DSP starts attempting to gain sync. To achieve this the DSP must continuously correlate the received data with the stored sync bits to attain a match of a high enough probability that it can safely assume that it has found the cell boundary. To achieve this it must first check every bit location in the radio frame for a match with the stored radio sync sequence and produce a likelihood value for each location being the cell boundary. After this initial routine the DSP then takes the location with the highest likelihood and makes the assumption that this is the cell boundary. It uses this location on the next radio frame to run the same routine. Once the likelihood reaches a certain level the algorithm assumes sync has been gained. From this point on instead of checking all radio frame locations every time it only checks 4 bits either side of its assumed cell boundary this will cover any phase shift in the received signal due to bit slip or loss. To do all 1324 radio frame bits every time is too processor intensive but it would be possible in future to carry this task out in hardware. When the radio sync has been achieved the cell is split and the payload is sent directly to the E1 controller DSP and the header is sent to the BCH decoder FPGA.

2) BCH Decode: All cell headers are transmitted to the FPGA including the idle cell headers that had been added by the VRIM controller on the transmit side. The FPGA uses the Berlekamp Massey algorithm [1] to detect and correct up to 3 errors in the header that may have occurred during the radio transmission. This algorithm has also been written in VHDL for ease of implementation in hardware. The process takes about 2 μ s to complete and hands the corrected cell header onto the E1 controller DSP.

3) E1 Controller (Receive side): The E1 controller software on the receive side removes the FEC extension and attaches the header back onto the payload. Here more bandwidth matching can be done if required, by the E1 controller board. The requirement will always be needed if the radio interface is less than 2Mbits/s. For example lets say that the data rate on the E1 input is 256Kbits/s and the rest of the cells are idle. All the idle cells would be removed by the cell sync routine on the input and the remaining data would be encoded. If the VRIM is configured for 512Kbits/s and connected to an equivalent radio then the VRIM controller DSP would be required to add idle radio frames to make up the bandwidth mismatch. On receiving this radio data the decoder would be running at 512Kbits/s and the E1 controller DSP would still be required to add idle cells to make the cell rate back up to the 2Mbits/s required by the switch.

Conclusions

The scheme we have designed in this paper has given us the ability to create a flexible architecture in which further more robust and complicated schemes can be developed. The environment has the ability to grow in size without any major problems as the schemes get more complicated. The scheme that has been described in this paper successfully passes data at 2 Mbits/sec in laboratory conditions without losing sync or ATM cells. The scheme has yet to be trialed with military equipment these trials will give us an idea of how good our chosen "proof of concept" scheme is. We do not believe that this scheme is the best scheme as management traffic in the payload will take hits on the radio link which could have affects on the network information used by the network management. Various schemes can now be implemented with a smaller design time and smaller engineering overheads because many of the integration problems have already been dealt with in this first testbed scheme.

References

- [1] Mapping Commercial ATM to the Tactical Radio Environment, R Barfoot, D Camm, J Daniel, P Thorlby, Defence Evaluation & Research Agency. Vol 3 pp 1055 – 1059 IEEE Military Communications Conference 1998.
- [2] Error Control Coding: Fundamentals and Applications, Shu Lin / Daniel J. Costello, Jr. Prentice-Hall series in computer applications in electrical engineering. 1983.

© British Crown Copyright 1999. Published with the permission of the Defence Evaluation and Research Agency on behalf of the Controller of HMSO.