SoC Integration of Reusable Baseband Bluetooth IP

Torbjörn Grahm
Ericsson Technology Licensing AB
Scheelevägen 15 , 223 70 Lund, Sweden
+46 46 193055
Torbjorn.grahm@ebt.ericsson.se

ABSTRACT

This presentation will give a list of design criteria an ASIC Design house need to look in the process of deciding to take the complex Bluetooth specification and implement everything from scratch or to integrate reusable Intellectual Property for integration into their SoC.

The presentation also include experience from a typical embedded development project where reusable Bluetooth Baseband Intellectual Property both for HW and SW is used with the Bluetooth Technology from Ericsson as example. This pper is a compressed summary.

1. INTRODUCTION

Bluetooth is a wireless short-link standard for 2.4 GHz twith feature for both voice, data and ad hoc networking capabilities that has hit the Electronics Industry development labs hard the last 2 years. The future will show that this complex piece of system on a Chip will be a standard-peripheral in a all future SoC:s. There is a lot of engineering effort put into various design-labs to understand how to implement the specification or how to get hold of the technology.

2. REUSABLE IP FOR BLUETOOTH

There are basically 2 choices to adopt new bluetooth technology into SoC-designs:

1. Read 1500 pages Bluetooth specification, hire a significant number of resources and implement both software and hardware parts from scratch, add the test-resources needed and insight in the qualification procedure and you might get a solution in-house.
2. Reuse existing complex IP-blocks for the Bluetooth functionality and evaluate the various solutions available.

This part is a summary of my experience in the ASIC-design reusability field and a framework for the task of analyzing the various parameters when selecting a solution for Bluetooth

2.1 Time Factors to evaluate

2.1.1 Time – Fast Learning Curve and availability

When integrating Bluetooth Technology it can be seen as done in various phases. **First** to evaluate the technology by use of Development Kits. **Secondly** integrate Compact Module with Firmware for interface over HCI (USB or Uart) ready for use together with a Host stack. **Finally** as reusable HW/SW Intellectual Property for integration into your SoC:s. For high volume, low cost devices the integration of Bluetooth IP in larger systems will become more attractive. To enable reuse a good training program is also required.

2.1.2 Time -Reuse defined HW/SW Interfaces

To enable an easy integration of Bluetooth IP it need to support some standardized hardware interfaces as a reference. For the software side well standardized and documented API:s give the same purpose for enabling a smooth software integration.

2.1.3 Time - reuse Development Tools

To reduce time in development projects including Hardware and Software development you need well proven, documented development Tools to speed up the time to working products and thereby have a stable Development environment to make your integration in.

2.1.4 Time – Additional Functionality Specification

With industry standards like Bluetooth, USB, GSM etc there will always be a roadmap of improved features within the specification like new profiles etc and also improved optimized functional blocks. The specification will be expanded over time and the solution selected need to be available as a roadmap of technology for the future needs.

2.2 Quality Factors to evaluate

When you make a design in-house you know at least the designers names and their skills. For external IP you rely on external know-how and the solutions can vary significantly in quality and perfornance. Potential Bluetooth Baseband IP functions can be divided into the below parts without closer defining if the function is implemented in Hardware or Software.

- Control-part: Generate all timing and state-control for the Blueototh control including Master Slave, Scatter Net support.
- Data-paths: Handle all the various data-packet types decode, encode, re-transmit for both Receive and Transmit Data packets.
- Voice-paths: Handle all the various voice-packet types decode, code, CVSD-coding, buffering and voice-format conversion for a certain number of simultaneous paths as defined in the Bluetooth specification.
- Link Manager IF and other CPU-load reduction blocks : This would be additional intelligence for reducing CPU-load and arrange buffering for various links as well as other parts that might be implementation specific to address certain system

DAC 2001, June 18-22, 2001, Las Vegas, Nevada, USA.

performance targets for the implementation such as low power, low CPU-load or what even the criteria can be.

2.2.1 Quality - CPU load-Interrupt , architecture

For various implementation the CPU-load can vary. Solutions in the Bluetooth industry vary between 1 MIP(s) up to more than 20 MIPS for a Bluetooth solution. First of all this will give a hit directly in the current-consumption parameter, but also it will give an indication if you can allow an embedded integration and using spare capacity of the CPU for applications with some real-time requirements.

There could also be solutions where more of the basic scheduling has hardware support and the interrupt issue is not that extreme. A basic parameter to bear in mind is that frame rate for Bluetooth link is 1.25ms for 1.0b specification, but for higher data-rates in 2.0-specification this might be a even faster rate to take care of. Software based solutions that should be low power also could have a problem with this. Blluetooth industry include solutions with a range of average interrupt rate of 1 per frame up to as less as 1 per 10 frames.

2.2.2 Quality – Voice, architecture

If you plan to include voice support in your Bluetooth solutions you should be aware of the following items. For very low power headsets with reasonable standby times the CPU-activity need to be low to get good performance. The various format conversions and CVSD-coding and additional potential digital filtering for being fit to integration into existing voice-links needs to be part of the solution.

2.2.3 Quality – Modularity -architecture

Bluetooth is a very complex and capable specification and not all applications will need the maximum performance that the specification allows. If the architecture is done scalable and modular this could be a bonus for your integration project to save silicon cost and RAM-memory area. The various tentative parameters to look into could be:

- Number of Voice Links
- Number of Supported Slaves
- Number of Supported Pico Nets.
- Type of Packets supported.

2.2.4 Quality – Solution Maturity and Test Methods

Bluetooth is a very young industry standard (1.0b specification released in 1999, SIG-group started in 1997). When analyzing this it should be known that Bluetooth is more complex than USB and in some areas even worse than GSM being only a point to point solution with fixed Base / Mobile.

All vendors will define that they have a fully proven system and passed un-plug fests etc, but it should be known that the only thing that is valid to enable putting a Bluetooth logo on the system is if the solution has passed the Qualification Rules defined by the SIG.

Internal development testing and methodology for that is key for the cases where some modifications to the IP are requested for integration purposes and a re-verification of the modifications planned.

2.3 Cost Factors to evaluate

Current consumption is seen as a Cost-factor in applications where Bluetooth Technology is used. When you make an assessment of the Current consumption figures you can use the below as one example to recalculate from:

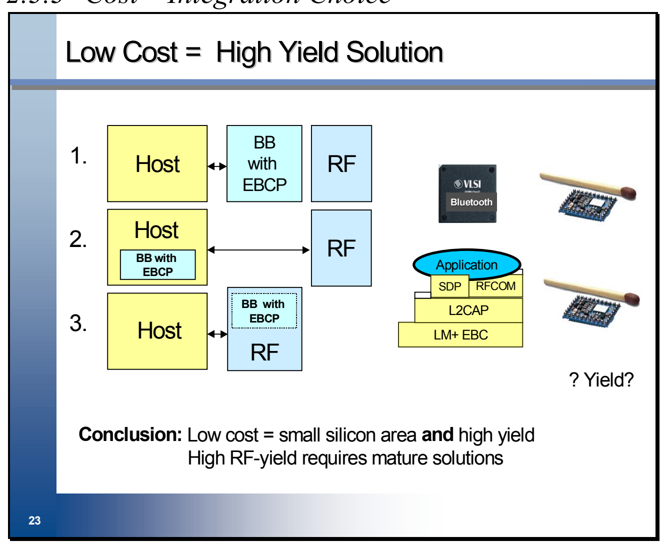
2.3.1 Cost - Current , Active Connection Data

An example for a data-connection with 10% duty cycle, a Radio-part active at 30 mA and Baseband active 4 mA will give a total average current consumption for this scenario of about 8mA and can be recalculated for your intended duty cycle and if other figures are used for radio and Baseband figures for various implementation. The range of solutions seen in the Bluetooth industry range for RF-parts between 30- 60mA and Baseband between 4 – 40 mA

2.3.2 Cost - Current , Standby Modes Scan

An example showing a Scan-scenario where a unit wake up once every 1.28 seconds to make a 11.25 ms scan activity. RF-part when active is about 30mA. Baseband core when running the Scan-procedure about 1mA. This scenario will then give an average RF+BB of 300uA and from that you can define the Standby time of your system when requiring the Scan-function switched on in your device.

2.3.3 Cost – Integration Choice



There are various integration options available when deciding how to Bluetooth enable your products.

1st case shows an add-on RF + separate Baseband interfacing over HCI to existing Host. This is a 2-CPU solution and can be done by existing components with the Ericsson IP included.

2nd case shows a case were the Baseband part (Baseband-IP) is integrated into the existing Host and thereby reducing the system cost. This can be a 2 CPU solution in one host or 1 CPU solution. The external RF-part can still be a well-proven RF-component from various vendors with Ericsson IP included.

3rd case show a case where the Baseband and RF-part of Bluetooth is integrated into one external device giving same functionality cut as in 1st scenario, but with less components.

All cases are possible and various solutions are available on the market, but to get lowest cost in case 3 you need to be aware of the yield items for RF and Baseband in one Die. In case 2 the cost can be reduced to less than case 3, but the risk and challenge is the Software integration.

2.3.4 Cost - Low CPU-load

CPU-load is a cost-factor, both for reducing current but also to enable smooth Embedded integration into your existing system. Increased complexity in Bluetooth functionality ranging from point-to-point data only up till full multi point with both voice and data-links functionality do not need to generate any increase in CPU-load if lot of hardware support is included, but with a Software based architecture you will definitely need to take a closer look to what happens when adding features in future product scenarios.

3. Bluetooth Enabling Project, a Case study

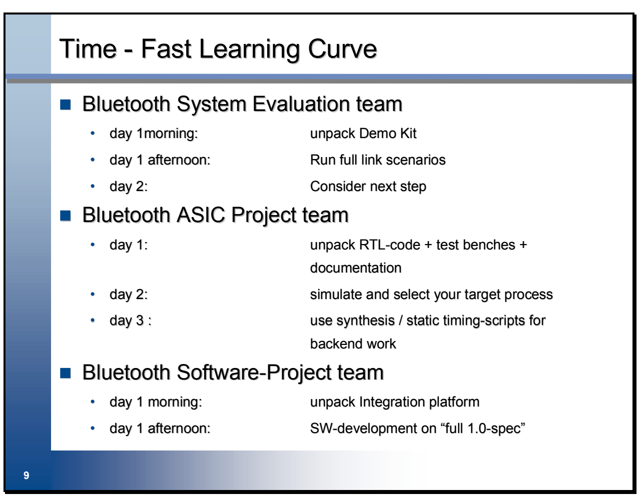
This section describe a case study for a typical Bluetooth enabling project where the main decision factors has been as below:

Table 1. Bluetooth Selection Criteria for this Design Project

Parameter	Comment
Time – Fast learning cure	Large amount of buy-in IP
Time – Reuse Development Tools and well defined interfaces	To reduce own design resourcing efforts
Quality- Mature Solution	To enable good interoperability with the bluetooth Standard
Quality – Complete solution	1500 pages of spec / profiles
Cost – Low CPU load	To enable embedded applications
Cost – Low Current Consumption	To enable handheld wireless devices.

With this case study the above objectives can be met and the below learning curve for the integration project can be adopted where we have used the product EBCP developed by Ericsson and ARM as a complete solution. There are basically 3 activities that needs to be enabled.

Table 2. Wanted time-plan with reusable IP.



3.1 System Team tasks and reuse items

The system evaluation team need to quickly get up the learning curve and also if needed do prototyping with standard component including the tentative IP-block. The system group also need to make the proper assessment of the quality of the IP-block. In this user-case the following items were used:

- Bluetooth Academy Training by Ericsson including basics about Bluetooth and the development Kits available.
- Bluetooth Development Kits for trials.
- Bluetooth Ericsson IP included in standard components for both Baseband and Radio components from any of the existing licensees.
- Bluetooth Ericsson Host Stack

3.1.1 Quality - CPU load , architecture issue #1

Based on the architectural definition in the EBCP-solution there is a Frame Scheduler controlled by Software with "look-ahead capabilities" that will enable a possibility to point at Chunk Descriptors and Receive Descriptors defining items to keep track on for each link. In bench-marking we have seen that this give a CPU-load of about 1.3 MIPS (Million Instructions per Second) at a ARM7TDMI and is roughly 5 times better than what has been seen in the Bluetooth industry so far. The benfits of this is shown in both lower Current consumption (less bus-activities etc), but also enables a smoother integration of this IP into existing ASIC-architectures that should become Bluetooth enabled.

3.1.2 Quality - Interrupts, architecture issue #2

Due to the architecture based on CD (Chunk Descriptors) and RD (Receive Descriptors) and the FS (Frame Scheduler) we can generate a system where we generate 2 interrupts per 14 Frames (1/7 Frames). This is about 7 times better that the Bluetooth Industry average and will also allow ups to keep the CPU load very low and also the current consumed. The other obvious benefit due to this is to enable simple integration of this EBCP into a System on a Chip already existing with lot of real-time constraints to be met.

3.1.3 Quality - Voice, architecture issue #3

Due to the hardware implementation of the Voice part in the EBCP we can run a voice connections continuously without any interrupts as long as there are not Management Packets to be sent. This is due to the built in DMA-support in the solution. The voice part includes support for 3 simultaneous voice-paths and voice-format conversion for these for various PCM-formats as well as CVSD. This hardware approach gives very low power consumption for voice, which is necessary for most headset applications.

3.1.4 Quality - Scaling, architecture issue #4

The reusable IP solution is scalable both for Hardware and Software.

•**PD:s (Piconet Descriptors)** defined between 0-4

•**CD:s (Chunk Descriptors)** defined between 1-10

•**RD:s (Receive Descriptors)** defined between 1-10

•**VD:s (Voice Descriptors)** defined between 0-3

•**FS (FrameScheduler length)** defined between 1-32

This enables en easy reduction of footprint in scalable way to fit the needs for the intended applications.

3.1.5 Quality - Maturity of Solution

Ericsson started already in 1994 with an internal small research project for short-link radios and in 1996 a first generation of Chipset for this was available. In 1997 the SIG was started and the specification was improved even further to become the Bluetooth Specification 1.0 (now 1.1) reviewed by the other SIG-members.

Ericsson 3rd generation Chipset is now one of the Blue Units used a reference kit for interoperability until the complete Qualification Procedure is in place. This generation will also be the basis for all 1st generation products like Mobile Phones, Headset and phone-plugs.

Ericsson 4th generation Chipset include full spec multi point etc and is also the basis for the reusable IP for HW SW that is used in this case-study, the EBCP-product.

3.1.6 Quality - System Test FPGA

To enable high quality testing in lab apart from the feedback from all Unplug fests within the Bluetooth community Ericsson used an approach of intensive FPGA-testing of HW + SW + RF in development platforms. 20 FPGA-boards each compliant with all features in the Bluetooth specification has been developed generating clusters of test-platform, some are point-to-point, some multi point.

A test specification covering all HCI-commands was developed and implemented also as automatic test-scripts testing all features intensively to cover all strange corner cases that might happen when running Bluetooth scenarios. Regression tests were made for each new Software release.

3.1.7 Summary – System Team Analysis

The above activities within the Bluetooth System Evaluation team made them prepared to decide that the EBCP-IP-block was mature and fulfilled their requirements and now it is up to the ASIC and Software team to get started with real development work.

3.2 ASIC Project Team Tasks and reuse items

3.2.1 ASIC overall architecture design

To enable easy integration of the EBCP-IP has adopted the AMBA 2.0 bus structure that is well proven for ARM-based systems For the Radio-interface the EBCP include support for BlueRF as well as several leading RF-vendors providing flexibility for vendor-choice.The first steps in the ASIC-design project is to get a complete overview of the intended block-functionality and how to integrate this into a complete System on a Chip (SoC). **EABBC Technical Reference Manual (SC039-DC-02001)** is the main reference source for the EABBC for use by both hardware and software engineers. It includes Introduction to the EABBC: its features, programmable parameters, interfacing diagram, Functional overview: block diagram, description of sub-blocks, description of operation, resets, clocks, initialization, operating modes, programming procedure and timing diagrams, Programmer's model for each block: register map, register descriptions, reset values, illegal values, equations to calculate programmed values, tables of example values and description of interrupts, Block I/O signal descriptions for each block: AMBA bus signals, on-chip signals and signals to device I/O pads.

3.2.2 HDL-design and functional simulations

When the architectural decisions are made you integrate you peripherals to the reusable Bluetooth IP functionality The **EABBC Integration Manual (SC039-DC-10001)** explains how to connect up the parts of the EABBC hardware description., how to get the test bench up and running in the simulator, how to run the integration tests, with the EABBC as supplied in the EBCP and how to add additional components to the hardware design.

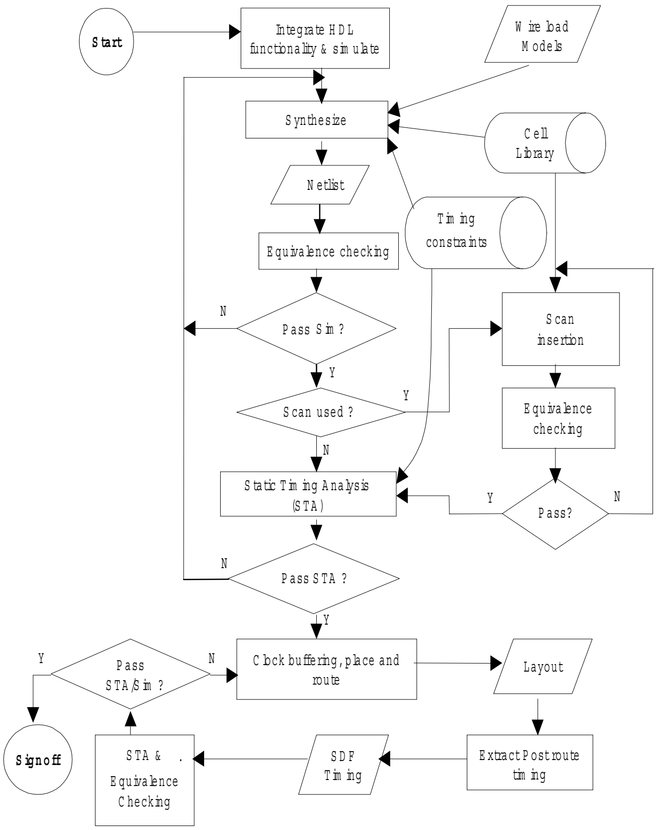
The delivered functional and behavioural VHDL-code consists of the following defined items. **EBC Synthesisable VHDL (SC039-MN-23001)** The EBC is the 'Enhanced Bluetooth Core' from Ericsson. This is an on-chip peripheral macro cell that performs the link control functions (including frequency hopping, data whitening. dewhitening, encryption/decryption etc) and includes a DMA controller. It is delivered in Synthesisable VHDL conforms to ARM's PrimeCell design guidelines for soft IP**. ARM Bluetooth MicroPack Synthesisable VHDL (SC039-MN-23002)** The synthesiseable VHDL of the ARM Bluetooth MicroPack **ARM Bluetooth MicroPack Behavioural Models (SC039-MN-23003)** The behavioural models of the ARM Bluetooth MicroPack, for simulation. **ARM Bluetooth Peripherals Synthesisable VHDL (SC039-MN-23004)** The synthesisable VHDL of the ARM peripherals for the Example AMBA Bluetooth Baseband Controller (EABBC) **ARM Design Simulation Model (AT010-MS-23402)** of the ARM7TDMI Rev 3 suitable for use with ModelSim VHDL simulator.

The above Functional blocks then needs to be simulated and assure that the integration into your ASIC has been done in the correct way before you go to the backend work.

3.2.3 3. Functional Simulation environment

The functional simulation environment and test-suite delivered consist of the **System Testbench (SC039-MN-23004)** that connects to the EABBC with simulation models to form a complete functional system in simulation. The connected system can then be used to run the tests defined in the **EABBC Hardware Integration Tests (SC039-VE-01001)** that consist of ARM source code, the purpose of which is to carry out integration testing of each part of the EABBC. These tests do not check the complete functionality of the EABBC but provide verification that interconnections of the blocks have been carried out correctly.

3.2.4 Back End ASIC Flow

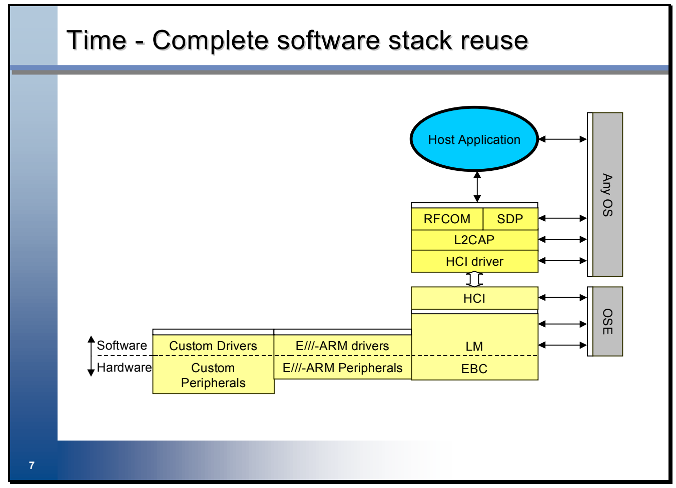


When you have a functional simulation working at RTL-level it is time to get the back-end flow iterations in place. All the relevant scripts for enable this are **Synthesis Scripts (SC039-MN-01001)** to synthesize the components and top level of the EABBC synthesizeable HDL to a netlist of gates from Avanti's CB25 cell library using Synopsys Design Compiler. **Static Timing Analysis Scripts (SC039-MN-01002),** Top level static timing analysis scripts for the EABBC for use with Synopsys Prime Time. **Formal Verification Scripts** for enabling a smooth comparison of RTL-level with Gate-level to avoid long timing simulations at gate-level for toll Chrysalis.

FPGA Synthesis Scripts (SC039-MN-01002) These scripts allow the synthesis of the EABBC Synthesisable VHDL into bitstreams suitable for loading onto the FPGA-Integrator platform

3.3 Software Team Tasks and reuse items

3.3.1 Documentation and Standard API:s



Normally there is a lot of pressure on the ASIC-team to deliver samples so the Software team can get stated in real-time environment, but in this case it is solved by getting the **Integrator Platform with BTLM** so ASIC-team is not out of the critical line for a while. Apart from the already mentioned EABBC Technical Reference Manual there exist also a **EABBC Software Reference Guide Reference (SC039-DC-02002)** for the EABBC Software that which parts of the Bluetooth Spec the software provides, exceptions and additions to the spec, supported platforms and processors. Functional overview: logical structure of software's modules and interfacing, description of each module, description of reset state, initialisation, operating modes, and configuration options. Configuration Model for each module: Configuration parameters, range of allowed values and their effect on functionality and performance.

3.3.2 Software Integration Components

EABBC BIOS Software (SC039-SW-02001) In order to ease the task of Licensee's software engineers when they add their own peripherals, or modify the ARM Bluetooth Peripheral, a set of BIOS calls and a set of drivers for the ARM Bluetooth peripherals, have been implemented. The EABBC BIOS software runs on the EABBC hardware and provides access to the EABBC hardware (excluding the EBC) for the EABBC Bluetooth Software and access to the EABBC hardware and OS resources for functionality added by the Ericsson Licensee.

EABBC Bluetooth Software (SC039-SW-02002) Software that runs on the EABBC and provides the following operational mode: HCI-LM – this provides the Link Manager, Host Controller Interface and UART Transport Layer suitable for implementing a Bluetooth Baseband controller.

3.3.3 System Integration Platform

In the EBCP case Ericsson teamed up with ARM to make the Bluetooth IP compatible to the Integrator Platform **Integrator/AP (SC039-BD-02003)** a standard ARM development board but modified to allow its correct operation with a **Bluetooth Logic Module (BTLM) (SC039-BD-02001)**– provides FPGA for EBC, and RF module and used together with **Integrator/CM7TDMI (SC039-BD-02002)** a standard ARM core module. While you are waiting for the ASIC-team to put the ASIC together (would take some weeks of production time, generally 6-8 weeks from Tape Out to sample and another 4-6 weeks before that to enable the backend flow meaning that you would have 10-14 weeks of time to spend on getting the complete system with application up and running in an emulated environment.

EABBC Software Integration Guide (SC039-DC-10002) describes the configurable elements of the SW to allow integration of the EABBC by the users end application.

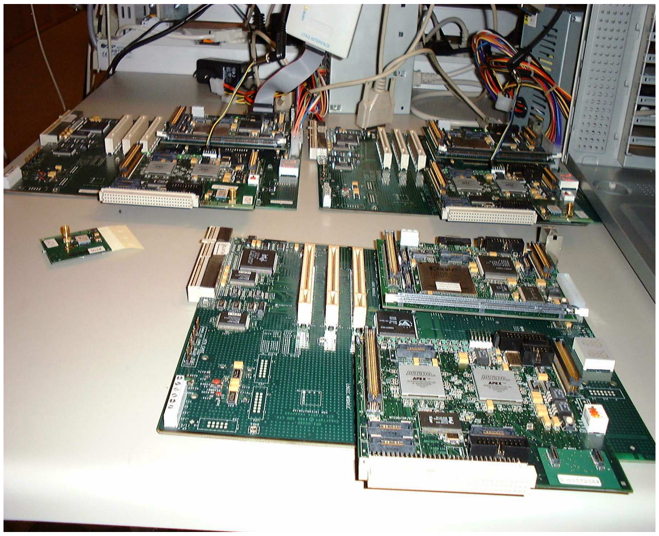


Figure 1 Integrator board with BTLM and RF-module

Bluetooth Development Platform User Guide (SC039-DC-10004) describes how to use the logic module (BTLM) in conjunction with an ARM integrator/AP development board including How to program the functional components of the EABBC into the FPGA's in the BTLM How to download the EABBC software onto Integrator/AP. How to execute and debug the EABBC software.

Bluetooth Test Software (SC039-SW-00003) also known under the name HCI Toolbox is a PC Software which can communicate with the EABBC through the UART Transport layer provided by the HCI-LM option of the EABBC software. This software normally runs on a PC with 2 serial ports – the first serial port (COM1) is connect to the first development board; the second serial port (COM2) is connected to the second development board. All relevant information about this is inlcuded in the **Bluetooth Test Software User Guide (SC039-DC-10003)**

4. CONCLUSIONS

An example Bluetooth enabling SoC-project with reuse-methodology has been presented together with the most important parameter to analyze to choose in-house design or IP-reuse. The conclusion is that reuse of both SW and HW is possible even with large IP-blocks if the complete design-process from fast-learning curve to final proven silicon is taken into account.

5. ACKNOWLEDGMENTS

Our thanks to the inventors of Bluetooth that have given us the opportunity for enormous engineering challenges to solve.

6. REFERENCES

[1] Bluetooth Official Web-site http://www.bluetooth.com .

[2] ARM Official Web-site http://www.arm.com