Software Defined FDD/TDD LTE Implementation on Sandblaster SB3500

Vaidyanathan Ramadurai, Sitij Agrawal, Saurabh Lahoti, Meng Yu ,Qingli Liu, Daniel Iancu, Gary Nacer, John Glossner
{vramadurai@optimumsemi.com}

ABSTRACT

In this paper, we present a software implementation of an LTE baseband receiver for both Frequency Division Duplex (FDD) and Time Division Duplex (TDD) system on Sandblaster SB3500 which is based on the Sandblaster 2.0 architecture.

The Sandblaster SB3500 is a compact and power efficient System on Chip (SOC) platform designed for wireless and multimedia devices. The SB3500 provides a high degree of parallelism both at the data level and at the instruction level. Additionally, the SB3500 allows simultaneous execution of 12 independent threads thereby allowing real time tasks to run in parallel. The SB3500 also has facility for different types of Direct Memory Access (DMA) like block DMA and scatter/gather DMA.

We provide an introduction to a LTE baseband receiver followed by a brief overview of the Sandblaster 2.0 [1] architecture. In LTE, the downlink multiple access is based on the Orthogonal Frequency Division Multiple Access (OFDMA) and the uplink multiple access is based on the Single Carrier Frequency Division Multiple Access (SCFDMA). We explore some key blocks in an FDD and TDD LTE system, difference in algorithms/implementation followed by thread level and block level optimizations for each system on SB3500 architecture.

Since most of the signal processing blocks are common between a TDD and an FDD LTE system, a single software code base for both the systems provides ease of maintenance and reusability. The software solution also provides easy upgradability to evolving standards.

1. INTRODUCTION

LTE is the next generation wireless OFDMA based technology ensuring higher throughputs and increased network capacity for increasing subscriber demands. In designing LTE, 3GPP committed to not only supporting FDD spectrum but also TDD spectrum [4]. TDD LTE was developed to take advantage of the several similarities with TD-SCDMA and also the existing technical advancements of FDD LTE. LTE FDD and TDD share the same underlying framework including radio access schemes, basic subframe formats etc. Both FDD and TDD LTE share the same set of specifications with few differences due to uplink/downlink switching. TDD LTE uses a single frequency sharing the channel between transmission and reception, spacing them apart by multiplexing the two signals on a real time basis. While FDD transmissions require a guard band between the transmitter and receiver frequencies, TDD schemes require a guard time or guard interval between transmission and reception. The time must be sufficient to allow the signal traveling from remote transmitters to arrive before a transmission is started and the receiver inhibited.

This paper is organized as follows. In Section 2, we provide an overview of an LTE baseband receiver blocks. In Section 3, we discuss the Sandblaster SB3500 multithreaded processor. In Section 4, we describe the difference in physical layer implementations of FDD and TDD LTE with a detailed description of our cellsearch implementation. In section 5 we discuss the implementation and multithreading of the LTE downlink receiver on SB3500 for real time performance. Section 6 concludes this paper.

2. LTE BASEBAND RECEIVER

2.1 CELLSEARCH

Figure 1 shows the LTE downlink receiver blocks. Cell search is performed by a UE when it powers on. The main purposes of cell search procedure are to acquire timing/frequency reference of the serving base station and its cell ID. The cellsearch procedure consists of the following:

- Locate OFDM symbol/frame boundary, detect the cyclic prefix length (normal or extended) and estimate/correct frequency offset between base station and UE.
- Obtain cell specific information, such as cell ID, system bandwidth and number of base station transmit antenna.

To assist cell search procedure, the base station broadcasts primary synchronization signal (PSYNC) and secondary synchronization signal (SSYNC) from one of the transmit antennas. When the synchronization signal is present, the occupied sub-carriers in other transmit antennas will be reserved and set to be zero. To support scalable

bandwidth from 1.4 MHz to 20 MHz, the synchronization signals are sent within 1.4 MHz bandwidth.

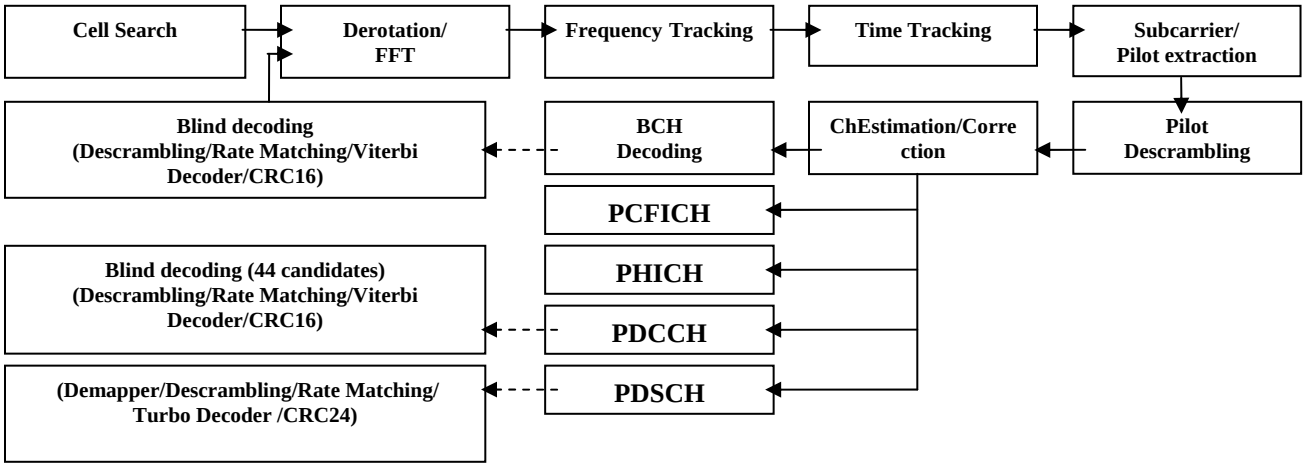

Figure 1 LTE Downlink Receiver

2.2 TRACKING

The tracking block gets raw data from the baseband IQ stream and runs once the cellsearch is complete. It is run continuously and is the first step of the downlink during steady state processing. The tracking algorithm is responsible for the following:

- Derotate the time domain input data based on the current frequency offset value, performing an FFT on the derotated data.
- Extract the subcarriers from the FFT results and pass the subcarriers to the next level of downlink processing.
- Update the frequency offset value through fine frequency tracking.
- Update the timing offset value through timing tracking and adjust for symbol boundary in time domain.

2.3 CHANNEL ESTIMATION

In LTE, channel estimation is pilot-assist-based. The pilot tones (or called reference signals in the specification document) are spread in the 2-D frequency-time plane for each transmit antenna. Not every OFDM symbol contains pilot tones. The pilots are more densely deployed in time domain for antenna port 0 and 1 than antenna port 2 and 3, implicating better channel estimates. For OFDM symbols which contain pilot tones, the pilots are equally spaced in frequency domain (one pilot per 6 sub-carriers).

At pilot sub-carriers, noise-corrupted raw estimates are obtained. From those raw estimates, channel estimation is performed for the rest of sub-carriers in the 2-D time-frequency domain. The calculated channel estimates are then used to equalize/correct the channel impaired data subcarriers.

2.4 BCH DECODING

The physical broadcast channel (PBCH) is the physical channel that carries the broadcast channel (BCH) transport channel. The BCH carries cell-specific information and is used for all types of UEs. Similar to the synchronization signals, the PBCH is transmitted in the center of the channel but it occupies 6 resource blocks (RB) or 72 subcarriers. The PBCH supports only QPSK modulation scheme and is located in slot #1 at OFDM symbols #0, #1, #2 and #3. The coded BCH transport block is mapped to four frames within a 40msec BCH TTI. Each frame is self-decodable and can be blind-decoded.

The baseband blocks as part of the BCH decoding are channel estimation, channel correction, QPSK demodulation, descrambler, rate matching, Viterbi decoder and CRC16.

2.5 PCFICH DECODING

The physical control format indicator channel (PCFICH) is the physical channel that carries the number of OFDM symbols used for transmission of PDCCHs in a sub-frame. PCFICH is located in OFDM symbol #0 of every sub- frame and the assignment to the sub-carriers is determined by the cell-id. The CFI channel coding is a block code with a coding rate of 1/16. The strong coding indicates the importance of this channel since any decode errors will result in failure to read PDCCH correctly. The

CFI can take values from 1 to 3. For transmission bandwidths of RBs more than 10, the control size is equal to CFI. Otherwise, the control size is CFI plus one.

2.6 PHICH DECODING

The physical hybrid automatic repeat request (ARQ) indicator channel (PHICH) is the physical channel that carries the hybrid ARQ indicator (HI). The HI contains the acknowledgement/negative acknowledgement (ACK/NACK) feedback to the UE for the uplink blocks received by the eNB. HI=1 is used for a positive acknowledgement and HI=0 is used for a negative acknowledgement. A simple repetition coding procedure with a coding rate of 1/3 is used.

2.7 PDCCH DECODING

The physical downlink control channel (PDCCH) is the physical channel that carries the channel allocation and control information. It consists of one or more consecutive control channel elements (CCE) where a control channel element corresponds to nine resource element groups. The number of OFDM symbols allocated to PDCCH is given by the control format indicator (CFI). The PDCCH supports only QPSK modulation. Multiple PDCCHs can be transmitted in a subframe.

The downlink control information (DCI) is mapped to the PDCCH in the physical layer. The DCI carries information regarding the following:

- Transport format information: modulation, coding scheme, redundancy version, and new data indicator, cyclic shift for demodulation, UL index, CQI request, downlink assignment index, HARQ process number, and code word information.
- Resource allocation information: RB assignment, hopping resource allocation, virtual resource block (VRB) assignment, HARQ information, Transmit power control (TPC) command.

The baseband blocks as part of the PDCCH decoding are channel estimation, channel correction, demapper, descrambler, rate matching, Viterbi decoder and CRC16.

2.7 PDSCH DECODING

The physical downlink shared channel (PDSCH) is the physical channel that carries the user data. The transport level data are mapped across different resource elements. The PDSCH supports QPSK, QAM16 and QAM64 modulation. The baseband blocks as part of the PDSCH decoding are channel estimation, channel correction, demapper, descrambler, rate matching, HARQ combining (for retransmissions), turbo decoder and CRC24.

3. SANDBLASTER DSP

The SB3500® features the Sandblaster® DSP for execution of baseband in software – including physical layer. It has a programmable RF interface, with the capability to capture raw data at 240 mega samples per second (MSPS). It includes interfaces to LCD, keypad, USIM, Smartcard, Audio codec, IrDA, plus emerging 'critical' features such as add-on memory cards, camera interface, and USB.

Sandbridge Technologies has developed the Sandblaster architecture [1] for convergence devices. As handsets are converging to multimedia multi-protocol systems, the Sandblaster architecture supports the data types necessary for convergence devices including RISC control code, DSP, and Java.

Figure 2 shows the architecture design of Sandblaster. The design includes a unique combination of modern techniques such as a SIMD Vector/DSP unit, a parallel reduction unit, and a RISC-based integer unit. Each Sandblaster core provides support for concurrent execution for up to four threads of execution. All states may be saved from each individual thread and no special software support is required for interrupt processing. The machine is partitioned into a RISC-based control unit that fetches instructions from a set-associative instruction cache. Instruction space is conserved through the use of compounded instructions that are grouped into packets for execution.

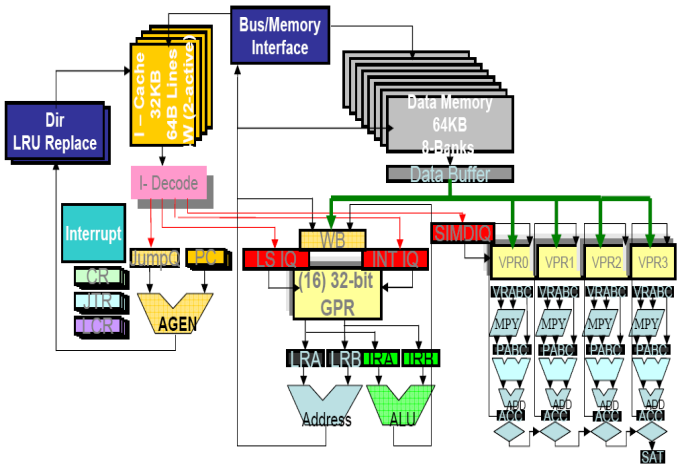

Figure 2 SB3500 DSP core

The memory subsystem has been designed carefully to minimize the power dissipation. The pipeline design in combination with the memory design ensures that all memories are single ported and yet the processor can sustain

nearly 16 taps per cycle for a filter (the theoretical maximum) in every thread unit simultaneously. A RISC-based execution unit, shown in the center of Figure 1, assists with control processing. In the case of control code, a 16 entry, 32-bit register file per thread unit provides for very efficient control processing. Common integer data types are typically stored in register files. This allows for branch bounds to be computed and addresses to be generated efficiently. The SIMD/Vector unit depicted on the right side of Figure 2 performs intensive loop processing. Each cycle, a 16x16-bit vector may be loaded into the register file while two vectors are being multiplied, saturated, reduced (e.g. summed), and saturated again. The branch bound may also be computed and the instruction looped on itself until the entire vector is processed. This may be specified in as little as 64 bits.

To enable wireless and multimedia processing in software, the processor supports several levels of parallelism. Thread-level parallelism is supported by providing hardware support for up to 4 independent programs to be simultaneously active on a single Sandblaster core. This reduces the latency in physical layer processing. Since many algorithms have stringent requirements on response time, multithreading is an integral technique in minimizing latencies. The data-level parallelism (SIMD) is supported through the use of a SIMD Vector unit. Additionally, the compound word instruction set provides instruction level parallelism.

The SB3500 consists of 3 Sandblaster DSP cores with an integrated ARM9 core, peripherals and external DDR memory as shown in Figure 3. A DSP core consists of 4 threads with each thread running at a frequency of 150MHz. Every core has its own local memory of 256Kbytes and an instruction cache memory of 32 Kbytes.

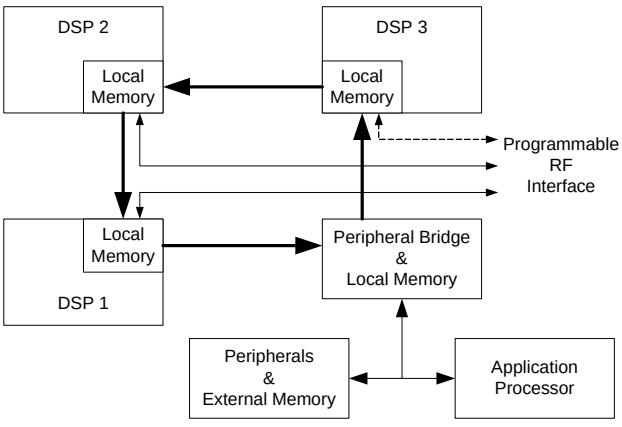

Figure 3 SB3500 Core

4. FDD/TDD LTE IMPLEMENTATION

In our SB3500 LTE design, we implemented the FDD physical layer and extended its support to TDD. There are no operational differences between a FDD and TDD modes in the overall system design and architecture. The key difference between the two systems stems from the placement of synchronization signals and support for different UL/DL allocation schemes in TDD mode. The difference in placement of synchronization signals impacts the cell selection procedure with a minor change in the control structure, which is discussed first. We will then describe the software implications of uplink downlink configurations and special subframes in TDD mode. Other features exclusive to TDD mode like short random access format, multi-subframe scheduling for uplink, acknowledgement bundling in uplink control channel and variable HARQ processes does not make a significant difference in the software implementation and will not be discussed in this paper.

4.1 CELLSEARCH

The cellsearch is the initial synchronization mechanism for LTE. It is responsible for finding the frame boundary, the cell id (0-503), the CP mode (normal or extended), and the initial coarse frequency offset.

The first step of cellsearch is to find the primary sync (PSYNC), which is located twice per frame in the last symbol of slots 0 and 10 for the FDD case and in the third symbol of slots 2 and 12 for the TDD case. Since this is the case the PSYNC is searched for within a half frame interval, which is equal to 9600 decimated samples or 75 symbols of length 128 assuming a sampling rate of 30.72MHz.

The PSYNC can be one of three sequences, so each of these three sequences must be cross-correlated with the input stream at each point. The length of each sequence is 128 (the number of decimated samples per symbol), and the total number of these cross-correlations to be computed is 3*9600 = 28,800. At each point the self-correlation is computed as well. As long as the cross-correlation over self-correlation is larger than THRESHOLD at any point for any of the three sequences, that information is saved. Once the entire half frame is searched all the saved points are checked and the “best” one is deemed the PSYNC. If there were no points that passed the threshold test the cellsearch returns that no PSYNC was found.

The coarse frequency offset is calculated by using the cross-correlation of PSYNC sequence with the input stream. Two correlations are computed, denoted ‘a’ and ‘b’. Correlation ‘a’ is computed as the correlation of the first half of the sequence (64 samples) with the first 64 samples in the input, and ‘b’ is computed as the correlation of the second half of the sequence with the next 64 samples in the input. Note that when we say input here we mean the start of the PSYNC in the input stream. The two correlations (which are complex numbers) are multiplied together, this

yields another complex number. The frequency offset is calculated as the arctan of the imaginary / real of the resulting complex number. Thus the angular change over 64 decimated samples is used as the basis for the frequency offset. Note that there are 1,920,000 decimated samples per second (from the conversions above, there are 19,200 decimated samples per frame), and 1,920,000 / 64 = 30,000. This means the theoretical maximum frequency offset that can be corrected is in a range of 30 kHz (+15 kHz to -15 kHz). In practice we find that a little more than half that range is acceptable, from + 8 kHz to -8 kHz.

The CP mode is determined in a similar manner as finding the PSYNC. At the start of PSYNC as found initially it is known that the CP is either the previous 32 samples (in the extended case) or the previous 9 samples (normal case). Furthermore since the CP copies the end of the symbol, the last 32 samples of the P-Sync symbol should correlate well with the CP in the extended case or the last 9 samples should correlate well with the CP in the normal case. Given all this information, the following algorithm is used: from the start of the PSYNC, 32 samples are counted backwards from the start and then the first 32-9=23 samples are used, call this block 'x'. That is, if i is the index of the start of the PSYNC, then block 'x' is [i-32 … i-10]. Similarly, 32 samples are counted backwards from the start of the next symbol (which is 128 samples after the start of the PSYNC) and then the first 23 samples there are used, call this block 'y'. Again using i as the start of the PSYNC, block 'y' is [i+96 … i+118]. Block 'x' is cross-correlated with block 'y', and block 'x' is also self-correlated. If cross-correlation over self-correlation is larger than THRESHOLD (not necessarily the same threshold as for PSYNC detection), then the CP mode is determined to be extended, otherwise it is normal. The reason for this is that in extended CP mode the 23 samples in block 'x' are actually part of the CP and should correlate well with the samples in block 'y', while in normal CP mode the samples in block 'x' belong to the symbol before the PSYNC and should have no relationship with the samples in block 'y'.

The final step of the cellsearch is to decode the secondary sync (SSYNC) and PSYNC to determine the cell id and to determine whether the PSYNC decoded is in slot 0/2 or 10/12 (this will tell us exactly where the frame boundary is located). The SSYNC is located in the symbol right before the PSYNC for the FDD case and three symbols before the PSYNC for the TDD case. Both symbols are derotated using the coarse frequency offset estimate. A 128 point FFT is performed on each symbol and the subcarriers are extracted. A channel estimate is formed by multiplying the PSYNC subcarriers with the conjugate of the FFT of the appropriate PSYNC sequence. This channel estimate is then applied to the SSYNC subcarriers via a multiplication. These values are then used to determine the cell id and slot number.

For PSYNC detection, each of the three saved reference arrays is cross-correlated with the input values, and this cross-correlation divided by the self-correlation is compared with a ratio to determine whether a PSYNC has been found. The equation for the ratio test is given as:

$$U(\vec{y}) = \frac{\left|\frac{1}{N}\sum_{i=1}^{N} y_i x_i^*\right|^2}{\left|\frac{1}{N}\sum_{i=1}^{N} y_i y_i^*\right|}$$

Where y is the input and x is the reference array. N, the length of the correlation, is 128 and the threshold value U is set to approximately 0.4 (note this can be changed according to system conditions, but for now this values is hard-coded as we shall see below).

One issue with this correlation is that the performance suffers when there is a large frequency offset. Using the two part correlation in the equation below performs much better in these cases (these are correlations 'a' and 'b' discussed above):

$$U(\vec{y}) = \frac{\left|\frac{2}{N}\sum_{i=1}^{N/2} y_i x_i^*\right|^2 + \left|\frac{2}{N}\sum_{i=N/2+1}^{N} y_i x_i^*\right|^2}{2\left|\frac{1}{N}\sum_{i=1}^{N} y_i y_i^*\right|}$$

Plugging in the values for N and U yields:

$$25.6 <> \frac{\left|\sum_{i=1}^{64} y_i x_i^*\right|^2 + \left|\sum_{i=65}^{128} y_i x_i^*\right|^2}{\left|\sum_{i=1}^{128} y_i y_i^*\right|}$$

So the two part cross-correlation divided by the self-correlation is compared with 25.6 (set to 25 in the code), any value greater indicated a potential match. For CP detection the one part correlation is used. In this case N is 23 and U is kept as 0.4, yielding:

$$9.2 <> \frac{\left|\sum_{i=1}^{23} y_i x_i^*\right|^2}{\left|\sum_{i=1}^{23} y_i y_i^*\right|}$$

As seen from above, the only difference between the FDD and TDD case for cellsearch purposes is the location of the P-Sync and S-Sync. In the FDD case, the P-Sync is located at the last symbol of slots 0 and 10, and the S-Sync is located in the second to last symbol of slots 0 and 10. In the TDD case, the P-Sync is located in the third symbol of slots 2 and 12, and the S-Sync is located in the last symbol of slots 1 and 11 (3 symbols before the P-Sync).

4.2 UPLINK DOWNLINK ALLOCATION

Table 1 shows the uplink downlink configurations in TDD mode. Each radio frame is of 10 ms duration and consists of 10 1ms subframes. Subframes 0 and 5 are always downlink subframes and contains broadcast information and synchronization signals. Subframe 1 is a special subframe as serves as a switching point from a downlink to an uplink subframe. Depending on the uplink downlink configuration, several other special subframes are inserted when there is a switching from downlink to uplink.

Uplink Downlink Allocation

UL/DL Config	Period (ms)	Subframe									
		0	1	2	3	4	5	6	7	8	9
0	5	D	S	U	U	U	D	S	U	U	U
1		D	S	U	U	D	D	S	U	U	D
2		D	S	U	D	D	D	S	U	D	D
3	10	D	S	U	U	U	D	D	D	D	D
4		D	S	U	U	D	D	D	D	D	D
5		D	S	U	D	D	D	D	D	D	D
6	5	D	S	U	U	U	D	S	U	U	D

Table 1. TDD LTE UL/DL Configuration

In FDD mode, all the subframes are downlink only as the uplink operates on a separate band. The software implementation of FDD frame processing is extended to TDD by having a control block that checks for the current subframe type. In FDD mode, this block always returns a downlink only subframe. The subframe length in terms of OFDM symbols is either 14 for normal CP or 12 for extended CP. In TDD mode, a table lookup returns the appropriate subframe type based on the current subframe number. A downlink subframe is similar to an FDD downlink subframe with either 14 or 12 symbols.

Special Subframe Configuration

Format	Normal CP			Extended CP		
	DwPTS	GP	UpPTS	DwPTS	GP	UpPTS
0	3	10	1	3	8	1
1	9	4		8	3	
2	10	3		9	2	
3	11	2		10	1	
4	12	1		3	7	2
5	3	9	2	8	2	
6	9	3		9	1	
7	10	2		-	-	-
8	11	1		-	-	-

Table 2. TDD LTE Special Subframe Configuration

A special subframe consists of 3 fields, Downlink Pilot Time Slot (DwPTS), Guard Period (GP) and Uplink Pilot Time Slot (UpPTS). Table 2 shows the different special subframe configurations. The length of each field is represented as a multiple of OFDM symbols. The physical layer keeps track of symbol timing and after getting the subframe type obtains the subframe length in terms of number of OFDM symbols. For a special subframe type, the GP and UpPTS symbols are skipped for downlink processing. The special subframe is treated like just another shortened downlink subframe with control information and reference signals.

5. DOWNLINK PHYSICAL LAYER IMPLEMENTATION

The LTE downlink physical layer is optimized and multithreaded to run on Sandblaster SB3500 platform. The building blocks are mapped to 6 threads or 1.5 DSP cores. Each thread runs at 150MHz. The thread partitioning is done as follows with each thread performing the functions listed.

- 4 threads for initial cellsearch and frame boundary acquisition. Once cell selection is complete, these threads are reused during steady state for tracking and symbol processing.

- 1 thread (shared with one of the cellsearch threads) for tracking
    1. FFT
    2. Pilot extraction and descrambling
    3. Frequency tracking and correction
    4. Time tracking and correction
    5. Subcarrier extraction

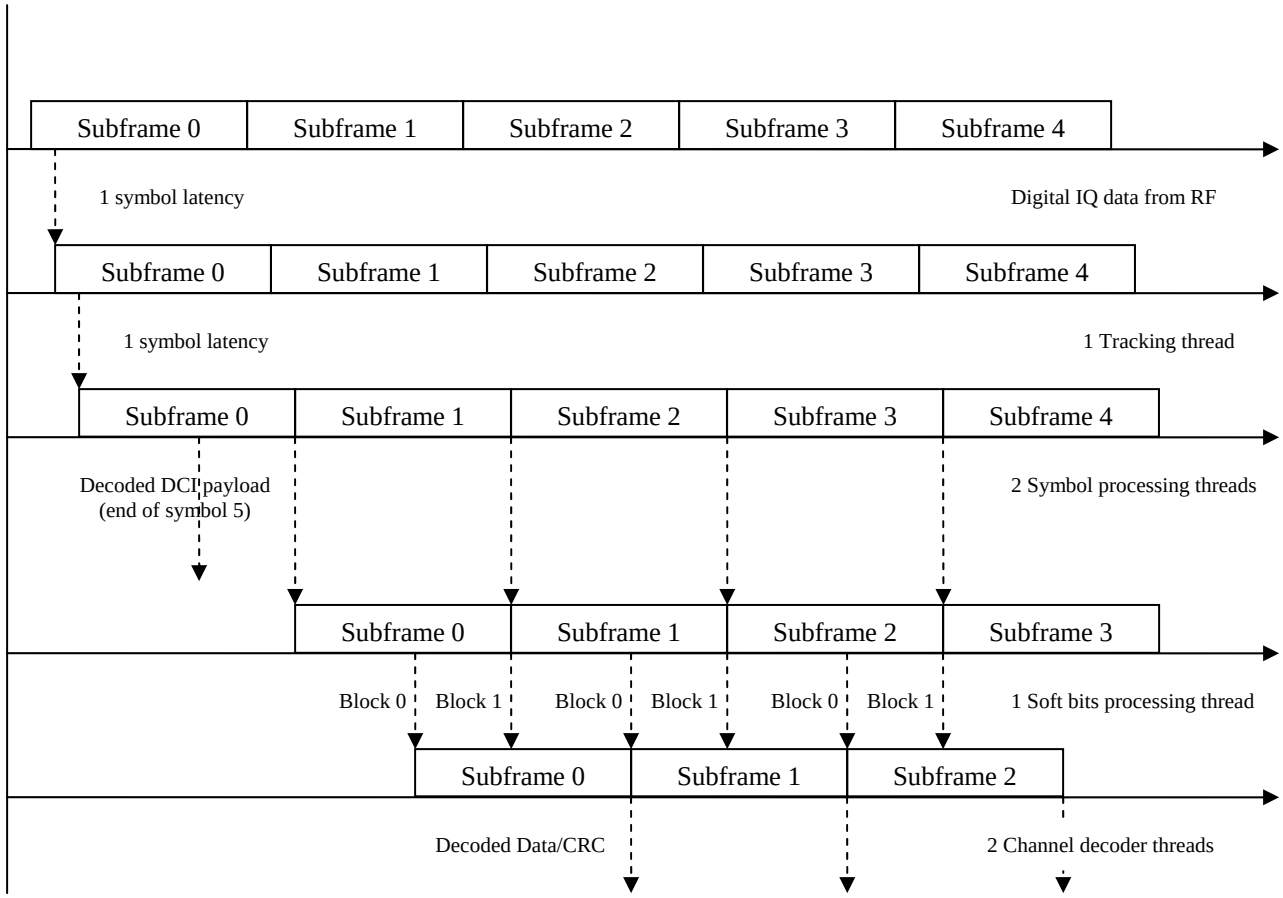

Figure 4 LTE Downlink Timing Diagram

- 2 threads for channel decoding
    1. Turbo decoder
    2. CRC24

- 1 thread (shared with one of the cellsearch threads) for symbol processing
    1. Channel estimation
    2. CSI estimation
    3. Channel correction
    4. Demapper
    5. BCH/PCFICH/PHICH decoding
    6. PDCCH blind decoding (1 DCI format)

- 1 thread for soft bits processing
    1. Descrambling of soft bits
    2. Rate Matching
    3. HARQ combining and restore

- 1 thread for PDCCH blind decoding (1 DCI format)
    1. Descrambling
    2. Rate matching
    3. Viterbi decoder/CRC16

Figure 4 shows the timing diagram of the downlink steady state process for a 2 block case. The tracking thread and the symbol processing thread work on a symbol level. The two threads are pipelined and synchronization is accomplished by dual buffering with shared memory. The symbol processing thread also performs PDCCH decoding for 1 DCI format (22 candidates) in addition to PCFICH and PHICH decoding. 1 thread is dedicated to perform PDCCH decoding for the 2nd DCI format (22 candidates). This thread works closely with the symbol processing thread for PDCCH decoding.

Synchronization is achieved by dual buffering the subframes using shared memory. The soft bits processing thread is pipelined with the channel decoder thread. Data sharing for synchronization is achieved by dual buffering using shared memory. 2 threads are used for channel decoding and CRC. The two channel decoder threads work cooperatively on the same input data. In other words, the multithreading is performed using data level parallelism.

Figure 5 shows the memory hierarchy for buffer management and synchronization in downlink receiver. The symbol, subcarriers and block buffers are stored in DSP local or L1 memory. The subframe buffers are stored in

external DDR memory. The tracking thread also gets a feedback from the BCH or symbol processing thread to update its bandwidth related parameters (e.g. N_RB). The symbol processing thread is pipelined with the soft bits processing thread on a subframe level.

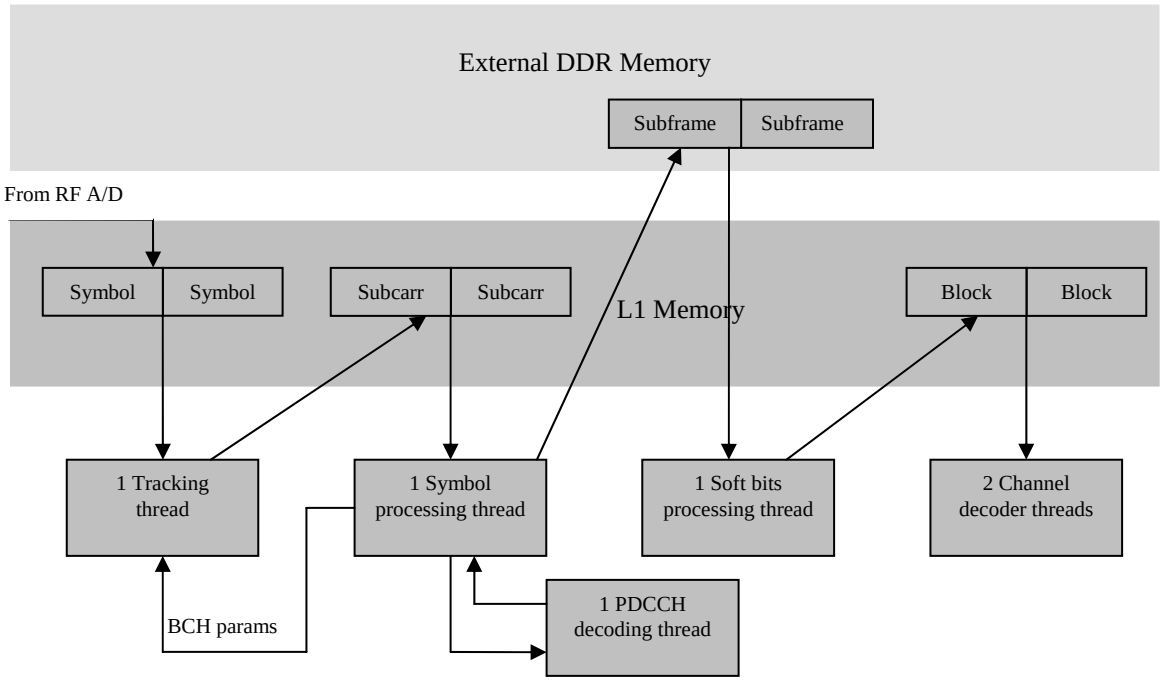

Figure 5 LTE Downlink Memory System and Synchronization

6. CONCLUSIONS

In this paper, we discussed a software defined implementation of an FDD/TDD LTE baseband receiver on Sandblaster SB3500. The key differences between a TDD and an FDD LTE system were highlighted and their implications on our software implementation were also described. We also described the optimizations and multithreading of downlink receiver blocks on SB3500. A software defined LTE solution provides ease of maintenance, reusability and flexibility to upgrade to newer evolving standards.

7. REFERENCES

[1] M. Moudgill, J. Glossner, S. Agrawal, and G. Nacer, "The Sandblaster 2.0 Architecture and SB3500 Implementation", in *Proceedings of the Software Defined Radio Technical Forum (SDR Forum '08)*, Washington DC, October, 2008

[2] J. Glossner and D. Iancu, "The Sandbridge SB3011 SDR Platform" *Symposium on Trends in Communications (SympoTIC'06), Invited keynote, Bratislava, Slovakia, June 24-26, 2006.*

[3] R. Ratasuk et.al, "TDD design for UMTS Long Term Evolution*", PIMRC-2008, Cannes, France, September 2008.*

[4] 3GPP LTE for TDD Spectrum in the Americas, November 2009

[5] John Glossner et al, "Sandblaster low power DSP", IEEE 2004 *Custom Integrated Circuits Conference, 2004, pp 575-581.*

[6] S. Jinturkar, J. Glossner, M. Moudgill, E. Hokenek, "Programming the Sandblaster Multithreaded Processor", *in Proceedings of GSPx 2003.*

[7] U. Ramacher, "Software-defined radio prospects for multistandard mobile phones, *Computer. Vol. 40, 10, pp 62-69, 2007.*

[8] A.Larmo et al., "The LTE Link Layer Design", *IEEE Commun, Mag, April 2009.*

[9] Zhenyu Tu, Meng Yu, Iancu, D. Moudgill, M. Glossner, J., "On the Performance of 3GPP LTE Baseband using SB3500", *International Symposium on System-On-Chip, 2009.*