ANALYSING SCHEDULABILITY OF SDR SYSTEMS BASED ON CYCLOSTATIONARY EXTENSION OF PERIODIC TASKS

Jan Westmeijer (mimoOn GmbH, Duisburg, Germany; e-mail: jan.westmeijer@mimoon.de) and David Guevorkian (Department of Signal Processing, Faculty of Computing and Electrical Engineering, Tampere University of Technology, Tampere, Finland; e-mail: david.guevorkian@tut.fi)

ABSTRACT

An important problem in designing SDR systems is to obtain minimal HW requirements to support implementation of desired set of radios. This problem can be solved by a method that would find out whether a valid schedule exists to perform, before predefined deadlines, all the tasks (algorithms) of a set of radios on given set of HW components. It is known that radios are periodic in nature (i.e. algorithms are periodically repeated). Therefore, to analyze schedulability of a single radio it is enough to consider only one period of that radio. However, since an SDR system must support several radios, larger period of time that includes several full periods of each radio must be analyzed. In this work, a method for analyzing availability of a schedule to perform given set of radios on a HW platform is proposed based on cyclostationary extension of radio tasks. A necessary condition for existence of a schedule is derived. This allows finding minimum HW requirements to support a desired set of radios.

1. INTRODUCTION

During the last decade, Software Defined Radio (SDR) attracts more and more attention of research community since it is thought to provide new, essentially wider possibilities to Communication Technologies [1] – [5]. An SDR system must support implementation of a set of radios each radio requiring implementation of a chain of baseband signal processing algorithms within very hard real-time limits. This brings a need for rather high computational power in SDR systems. On the other hand, SDR devices should satisfy to strict power consumption and pricing constraints. Therefore, in designing an SDR system, naturally a task arises to find the minimal amount of HW resources that is enough to support given set of radios.

In an SDR system, the digital baseband signal processing is divided into software tasks that are scheduled by a scheduler and carried out by underlying processing elements of shared HW platform, *e.g.* processors and hardware accelerators. To make an SDR system efficient, the set of radios should share the HW resources as much as possible. Due to limited processing capacity and power consumption budget in devices, an efficient schedule is of crucial importance for the overall system performance and hence Quality of Service for the user.

In contrary to the traditional ASIC approach where each radio baseband has its own dedicated signal processing hardware, the SDR approach is based on radio-independent signal processing resources *e.g.* Vector Processor or DSP that are shared by a subset or all of the currently running radios. All signal processing tasks have to be assigned to one of the processing elements within the constraints given by their required processing time, dependencies to other tasks and absolute deadline set by the radio standards.

Another limiting factor for the search of optimal schedule is the processing time that may be devoted to finding the scheduling algorithm. The optimal scheduling of parallel tasks with some precedence relationship onto a parallel machine is known to be NP-complete and hence the processing time consumed by an exhaustive search for the most optimal schedule is unacceptable in an operation environment, *e.g.* radio baseband with clear real-time requirements. Therefore, more sophisticated scheduling approaches are to be used in practice [7]-[9].

For example, in online dynamic scheduling approaches, the scheduling for signal processing tasks is done when a task is activated during run-time. The scheduler is not aware of the periodicity of the radio baseband and hence is unable to predict coming tasks beyond the currently running tasks. This implies that instructions have to be loaded into local memory of the processing elements on-the-fly, which would jeopardize the tight timing constraints in the baseband domain in most cases.

Another approach is the static scheduling approach where, for each radio baseband, time slots on processing elements are reserved for each signal processing task during design time (compile time). Therefore, the reserved time slots cannot be changed according to the dynamic run-time

situation resulted, *e.g.* from different combination of radios running in parallel. This leads to suboptimal schedules and as result, to over-dimensioned platforms.

In [9], a scheduling approach is proposed where static schedules are dynamically created for all possible radio combinations, to which the system may arrive after the moment when the schedulers are designed. Once the system changes its state (combination of currently active radios), the corresponding schedule is readily available and may be used. This significantly relaxes the time constraints for schedule design.

In this work, we propose a new method of obtaining minimal HW resources for given set of periodic jobs (radios) based on a new method to analyze and determine the schedulability of cyclo-stationary software task sets on given set of HW components, *e.g.* schedulability of baseband signal processing in concurrent operation of multiple radios on a HW platform consisting of several types of processing elements. The proposed method allows using HW platforms with minimal resources really necessary to support implementation of a given set of radios. The method is independent of the actual scheduling policy e.g. Earliest Deadline First (EDF), Rate Monotonic (RM) or Deadline Monotone (DM), *etc*.

2. CYCLOSTATIONARY EXTENSION OF A SET OF PERIODIC JOBS

In this section we present a timing diagram based model of periodic tasks, *e.g.,* radios. This allows deriving a formal criteria to analyze schedulability of periodic task sets. In particular, in the next section, based on the presented model we derive a necessary condition for existence of a valid schedule to implement a set of periodic tasks on a computing platform containing several sets of processing elements. First we present timing based diagram of a single periodic task (i.e. radio) in Section 2.1, then we present cyclostationary extension of periodic tasks (radios) and timing diagram based model of a set of such tasks in Section 2.2.

2.1. Timing based model of a single periodic task

Let the following set of time intervals represent the timing information of m_1 algorithms in a single radio R^1:

$$S^1 = \left\{ A_0^1, A_1^1, ... A_{m_1}^1 \right\}, \quad A_k^1 = \left[t_k^1, t_k^1 + w_k^1 \right] \qquad (1)$$

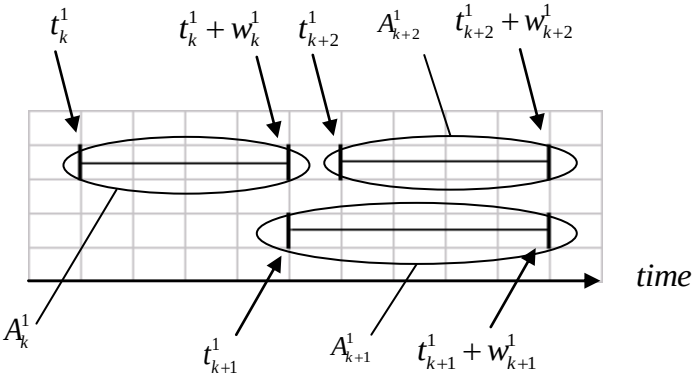

Figure 1. Execution time intervals of tasks.

where $k = 0,...,m_1$, $t_k^1 \in R^+$ is a positive real number indicating the starting time of k^{th} algorithm A_k^1 relative to the initial time t_0^1 of radio R^1, $t_k \le t_{k+1}$, $k = 0,...,m-1$, $w_k^1 \in R^+$ is the execution time of the k^{th} algorithm, and $m_1 \in N$ is a positive integer. The starting time of an algorithm is defined as the earliest possible time to start it. The execution times are measured with respect to a ''Reference'' Processing Element (PE) having unity processing power.

As an example, Fig. 1 illustrates the timing representation of a radio described by (1) a radio is represented as a chain of algorithms wherein the precedence dependencies of algorithm are implicitly described. Note that the set S^1 is ordered in the sense that $t_k \le t_{k+1}$, $k = 0,...,m_1 - 1$. On the other hand, $A_k^1 \cap A_j^1$ does not have to be empty for any $k, j \in 0...m_1 \wedge i \neq j$ and, therefore, equation (1) does allow algorithm concurrency.

For example, Algorithms A_{k+1}^1 and A_{k+2}^1 on Fig. 1 are implemented concurrently. As shown in Fig. 1, none of the algorithms of radio R^1 is implemented prior to time t_0^1. Iimplementation of the Algorithm A_k^1 of radio R^1 starts at time t_k^1 and ends at time $t_k^1 + w_k^1$. Algorithm A_{k+1}^1 of radio R^1 is implemented within the time interval between t_{k+1}^1 and $t_{k+1}^1 + w_{k+1}^1$. The implementation tome of the Algorithm A_{k+2}^1 of radio R^1 is the interval between t_{k+2}^1 and $t_{k+2}^1 + w_{k+2}^1$. As shown in Fig.1, implementation of the Algorithm A_{k+2}^1 may be started before the implementation of Algorithm A_{k+1}^1 is complete. In this regard, Algorithms A_{k+1}

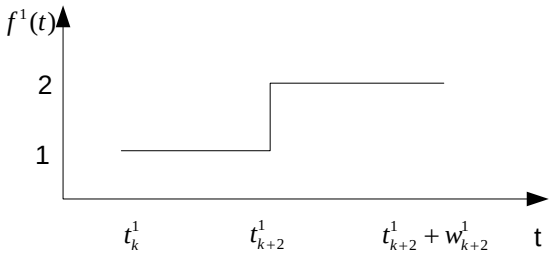

Figure 2. Execution time intervals of tasks.

and A_{k+2}^1 are independent of each other but may only be dependent on the preceding algorithms.

Once time intervals (1) are known, a radio, from implementation point of view, can be modelled as:

$$f^1(t) = \sum_{k=0}^{m_1} \Delta_{A_k^1}(t), \ \Delta_{A_k^1}(t) = \begin{cases} 1 & t \in A_k^1 \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

where, in fact, $f^1(t)$ indicates the number of algorithms that are concurrently implemented at time instance t. For instance, the function $f^1(t)$ for the radio that corresponds to the timing diagram of Fig. 1 is given by (see also Fig. 2):

$$f^1(t) = \begin{cases} 1, \text{ for } t \in [t_k^1, t_{k+2}^1] \\ 2, \text{ for } t \in [t_{k+2}^1, t_{k+2}^1 + w_{k+2}^1] \\ 0, \text{ otherwise} \end{cases}$$

since, according to Fig. 1, at each of time intervals $[t_k^1, t_k^1 + w_k^1]$ and $[t_k^1 + w_k^1, t_{k+2}^1]$ only a single algorithm (A_k^1 or A_{k+2}^1, respectively) is implemented, at time interval $[t_{k+2}^1, t_{k+2}^1 + w_{k+2}^1]$ two algorithms A_{k+1}^1 and A_{k+2}^1 are implemented, and no algorithms are implemented at other times.

2.1. Cyclostationary extension of periodic task sets

Radios are periodic in nature. Therefore, one can consider cyclic or repetitive extension of $f^1(t)$ with a period T^1

$$f_r^1(t) = f^1(t + a \cdot T^1), \quad a \in N \quad (3)$$

as depicted on Fig. 3. The period T^1 may represent, for example, one or several OFDM symbol(s) or any other natural sized packet duration. Although **Figure 3** does not show overlap between the periods, this is not excluded.

For example, in IEEE 802.11a WLAN, the same chain of algorithms is repeated for each OFDM symbol having duration of four microseconds ($4\mu s$). Therefore, it is natural to consider cyclostationary extension (3) of that chain of algorithms with the period of $T^1 = T^{WLAN} = 4\mu s$. In another example, considering 20MHz 3GPP LTE E-UTRA implementation, a natural choice of the period in (3) would be $T^1 = T^{LTE} = 1ms$ (one millisecond), which is the duration of one OFDM sub-frame, since in this radio, all the computations can be arranged to be repeated for each sub-frame.

So far we considered implementation of a single radio. Let us now consider the case where several radios should be implemented on top of shared HW resources. Similarly to radio R^1, the i^{th} radio R^i, $i = 1,...,n, n \in N$, can be defined as

$$f^i(t) = \sum_{k=0}^{m_i} \Delta_{A_k^i}(t)$$

having period T^i, and time intervals $S^i = \{A_0^i, A_1^i, ...A_{m_i}^i\}$, $A_k^i = [t_k^i, t_k^i + w_k^j]$. Assuming periods T^i be natural numbers in ns, μs, or say in ms, let us consider their Least Common Multiplier (LCM):

$$T_{lcm} = \frac{T^1 \cdot ... \cdot T^n}{\text{GCD}(T^1 \cdot ... \cdot T^n)} \quad (4)$$

where GCD stands for Greatest Common Divisor.

Concatenating each set of intervals S^i, $i = 1,...,n$, (T_{lcm} / T^i) times, we get

$$S_r^i = \left\{ A_0^i, A_1^i, ... A_{\left(\frac{T_{lcm}}{T^i} - 1\right)T^i + m_i}^i \right\},$$

which can be interpreted as the cyclic extension of radio $f^i(t)$. Let us note that presented in this way, all the radios will have the same period T_{lcm}.

As an example, let us assume we need to create a scheduler for an SDR system that supports implementation of IEEE 802.11a WLAN and 20MHz 3GPP LTE E-UTRA. In that case, $T_{lcm} = 1ms$, which is the period of the whole algorithm chain for both radios. This means that if a valid schedule is created for one period of time $T_{lcm} = 1ms$ it can then be repetitively applied during the whole time when both radios need be operated. Therefore, schedulability analysis may be reduced to analysis of only one period. Note that during one period one LTE OFDM sub-frame and *250* WLAN OFDM symbols are processed simultaneously.

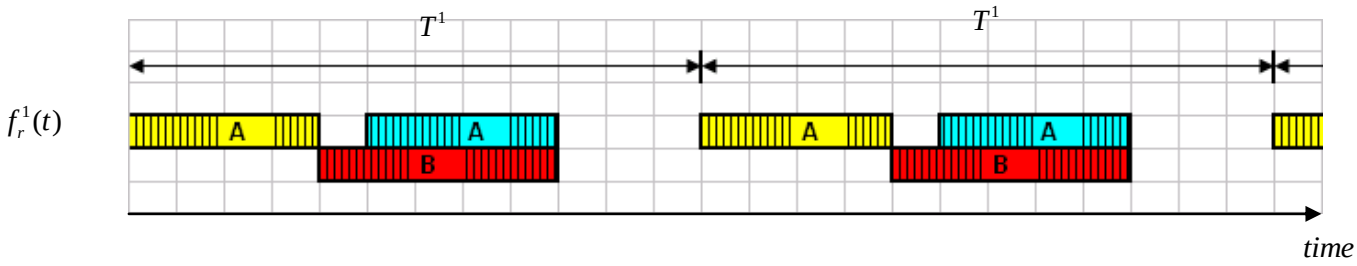

Figure 3. Periodicity of radio algorithms

The sets S_r^i, can be united and ordered to a single set S to correspond to the combination of radios R^i, $i=1,...,n$:

$$S = S_r^1 \cup ... \cup S_r^n \tag{5}$$

Now, it is easy to see that the combination of radios R^i, $i=1,...,n$, is periodic with the period T_{lcm} and the total processing time T_p needed to implement this combination of radios within one period can be computed as

$$T_p = \sum_{i=1}^{n} \frac{T_{lcm}}{T^i} \sum_{j=0}^{m_i} w_j^i$$

3. NECESSARY CONDITION FOR SCHEDULABILITY OF A RADIO SET

Suppose now that a set of radios must be implemented on a HW platform consisting of K identical "reference" processing elements (PEs) each having unity processing power. Then the total available processing power within one common period T_{lcm} could have been measured as $T_{\text{ap}} = K \cdot T_{lcm}$ the most that corresponds to the ideal case where it was possible to achieve 100% utilization of all PEs during whole processing. Therefore, the following term

$$\eta = \frac{T_p}{T_{ap}} = \frac{\sum_{i=1}^{n} \frac{T_{lcm}}{T^i} \sum_{k=0}^{m_i} w_k^i}{K \cdot T_{lcm}} = \frac{1}{K} \sum_{i=1}^{n} \sum_{k=0}^{m_i} \frac{w_k^i}{T^i} \tag{6}$$

may be used to derive a necessary condition for schedulability of radio combinations on HW platforms. Namely, one can state that the set of radios $\{R^i, i=1,...,n\}$ cannot be implemented on the system with a set of K "reference" PEs if η defined in (6) is larger than unity ($\eta > 1$).

This necessary condition can easily be generalized to the case where the system includes PEs of different types. Without loss of generality suppose the system contains PEs of two types. In this case, the algorithms of radios are split into two clusters I and II depending on the type of PE where they are implemented. Now instead of the measure in (6) the following measure may be used:

$$\eta' = \max\left\{\frac{T_p'}{T_{ap}'}, \frac{T_p''}{T_{ap}''}\right\} \tag{7}$$

where T_p^I and T_p^{II} are required processing times of algorithms from cluster I and II on corresponding types of PEs, and T_{ap}^I and T_{ap}^{II} are available processing times provided by PEs of type I and II, respectively. The terms T_p^I, T_p^{II}, T_{ap}^I, and T_{ap}^{II} are calculated similarly to corresponding terms in (6). The necessary condition now transforms to $\eta' > 1$.

Let us note that the necessary condition based on (7) assumes semi-static task assignment where the decision on which algorithm to implement on which type of PE is made at compile time (that is before the actual implementation starts) and remains fixed during the whole execution. This does not necessarily mean fully static assignment since only type of PE is decided at compile time but the exact PE assignment may be implemented dynamically. Also, criterion (7) corresponds to the case where the platform comprises of only two types of PEs.

In a more general scenario, where the platform may consist of several types of PEs, say PE clusters $PE^{(r)}$, $r=1,...,l$, and where fully dynamic task assignment is allowed meaning that the algorithms are assigned to PEs during the execution time, the equation (7) should be generalized. In this case, (7) transforms to

$$\eta'' = \min_{\substack{\text{all possible} \\ \text{assignments } a}} \left\{ \max_{r=1,...l} \left\{ \frac{T_p^r(a)}{T_{ap}^r} \right\} \right\}, \tag{8}$$

where $T_p^r(a)$, $r=1,...,l$, is the total processing time of all algorithms assigned to PEs from cluster r according to assignment a, and T_{ap}^r is available processing time or power provided by all the PEs of that cluster. In this scenario, the necessary condition transforms to $\eta'' > 1$.

The necessary conditions based on (6) - (8) are valid but are rather weak since they do not take into account data transfer times. When taking these data transfers into account, (6) is transformed to

$$T_{p+b} = \sum_{i=1}^{n} \frac{T_{lcm}}{T^i} \sum_{j=0}^{m_i} (w + t\_bus)_j^i \qquad (9)$$

where $(w + t\_bus)_j^i$ represents the execution of the i^{th} algorithm including data bus transfer time. If the ratio

$$\frac{T_{p+b}}{T_{av}}$$

is larger than unity, the radios may only be, in the best case, implemented with a limited amount of data transfers between PEs. In practice, this determines which algorithms have to reside on the same processing element.

It should be noted that the derived necessary conditions do not generate or guarantee the existence of a valid schedule. For example, if we take sufficiently large number *K* of PEs, we may make the value of η less than unity. However, this would not mean a valid schedule is possible since there may be algorithms that cannot be implemented within a pre-specified time irrelevant on how many PEs are used. The algorithm may be essentially sequential or it may require too extensive data communication when parallelized to several PEs. The proposed necessary conditions do not reveal such situations and, therefore, cannot be considered as satisfactory conditions. However, the introduced model and the derived necessary conditions are useful for schedulability analysis and for the design of an SDR system with minimal HW resource since they indicate (to designers or to automated schedule creating systems) the maximum number of PEs that is necessarily needed to support implementation of a specified set of radios.

4. CONCLUSION

Timing diagram based model of radio sets was introduces. This model explicitly uses periodic nature of radio tasks. In particular it formally defines the minimum period of time for which valid static schedules for a set of radios should be designed and verified or simulated. Also based on the proposed model, necessary conditions were derived for existence of a valid schedule to implement a set of radios on a given SDR platform. The derived necessary conditions do not guarantee existence of a schedule for mapping the considered combination or radios onto a given HW platform. Neither, they provide a valid schedule. Nevertheless, these conditions may be used to analyze whether given set of radios is , in principle, possible to map onto a given HW platform, thus simplifying the design of SDR systems with minimal HW resources.

5. REFERENCES

[1] A. Ahtiainen, H. Berg, U. Lücking. A. Pärssinen. and J. Westmeijer, "Architecting Software Radio," *Proceeidngs of the SDR 07 Technical Conference and product Exposition*, 2007.

[2] A. Ahtiainen, K. van Berkel, D. van Kampen, O. Moreira, A. Piipponen, T. Zetterman, " Multi-radio Scheduling and Resource Sharing on a Software Defined Radio Computing Platform," *Proceeidngs of the SDR 08 Technical Conference and product Exposition*, 2008.

[3] L. Harju and J. Nurmi, "Hardware platform for software-defined WCDMA/OFDM baseband receiver implementation," *IET Comput. Digit. Tec.*, Vol.. No 5, pp. 640-652, 2007.

[4] F. Kasperski, O.Pierrelee, F. Dotto, M. Sarlotte, "High data rate fully flexible SDR modem advanced configurable architecture & development methodology," *Proceedings of Design, Automation & Test in Europe Conference & Exhibition, 2009. DATE '09,* pp. 1040-1044, 2009.

[5] H. Berg, C. Brunelli and U. Lücking, "Analyzing models of computation for software defined radio applications," *Proceedings of International Symposium on System-on-Chip, 2008 (SOC-2008)*, pp. 1-4, 2008.

[6] S. Sriram, Sh. S. Bhattacharyya, *Embedded Multiprocessors. Scheduling and Synchronization.* Signal Processing and Communication Series, Marcel Dekker Inc., 2000, 327.

[7] O. Moreira, F. Valente, M. Bekooij, "Scheduling multiple independent hard-real-time jobs on a heterogeneous multiprocessor," *Proceedings of the 7th ACM & IEEE international conference on Embedded software. 2007*, pp. 57-66. 2007.

[8] O. Moreira J.-D. Mol, M. Bekooij and J. van Meerbergen, "Multiprocessor Resource Allocation for Hard-real-time Streaming with a Dynamic job-mix," in *Proceedings of IEEE Int. Symposium Real Time and Embedded Technology and Applications (RTAS-2005)*, pp. 332-341, 2005

[9] D. Guevorkian, J. Westmeijer, "Predictive scheduling of job combinations in SDR systems," submitted to SDR'11 WInnComm, 2011.