Synthesis of Complicated Asynchronous Control Circuits Using Template Based Technique

Sufian Sudeng and Arthit Thongtak

Abstract— **this paper proposes an approach for complicated asynchronous controller synthesizing. The proposed scheme introduces a template based technique in state encoding process to insert additional signals to STG with a small number. According to our method, complete state coding (CSC) property can be satisfied without using state graph tracing. Our method is useful for complicated asynchronous controllers, and also it can guarantee the other relevant properties, such as persistency and consistency. Our process begins with an encoding STG using Petri-net level in order to form a template STG. Then the projection to each non-input signals from an original STG is done. After that, we trace the projection to smaller state space. If the small state space shows conflicts, we have to insert balance signals from template STG. Unbalance signals are inserted after in case the state space still shows conflicts. Finally, we can get the STG with appropriate insertion points which is used to be projected for CSC support on each non-input signals. Asynchronous DMA controller is an example of our proposed method. The final part of this paper is concluded with a complexity comparison between our template based method with state based method and structural encoding method. It shows that the number of iterative signal removal according to our method is less than others.**

***Index Terms*— Signal Transition Graph (STG), Logic Synthesis, Asynchronous Control Circuits, Asynchronous DMA Controller, Template Based Technique.**

I. Introduction

State encoding is a process in logic synthesis for asynchronous control circuits. It is defined as a given specification to interact between input and non-input signals. The target is to find an encoding for every state of non-input signals in order to implement the circuits with hazard free. To find a state encoding, several authors have concerned with different ways, such as a logic synthesis using Petri-net unfolding synthesis using state based technique [1,2,3], synthesis using structural encoding technique [6], and etc.

State based technique is involved several problems. The major problem is the detection of state encoding conflicts. It causes exponential complexity of conflict states with size of specification.

Another problem is the insertion of additional signals to solve the conflict states. Signal insertion must be done in such a way that the new signals are consistent (rising and falling transitions alternate) and their behaviors are hazard free. This method is insufficient for the specification such large scale of signals. Structural encoding method is more suitable than classical state based technique. The structural encoding scheme is Petri-net level encoding. This method encodes with two silent transitions [3,6] to generate encode STG. Signal removal is begun after encoded STG has been generated. Iteratively removing signal is done with greedy removal from encode STG. In the end, it is produced a reduced STG. The next is to start projection for CSC support for each non-input signals. This method is available for large state encoding. However, it suffers from complexity reduction in greedy removal phase according to the number of iterative signal reduction.

This article proposes a template based method for synthesizing complicated STGs, and it also useful for some large scale STGs. The template based technique not only be useful for complicated and large scale STGs but also solves the reduction complexity of structural encoding technique.

Our template based technique was explained by one complicated example called asynchronous DMA controller that has been used in our asynchronous processor implementation [7].

We introduced three types of synthesis scheme in our explanation. Firstly, the synthesis of asynchronous DMA controller using state based technique. We derived the DMA's STG to state graph and showed the limitation of synthesizable properties which is limited by STG's rule called complete state coding (CSC). The complete state coding meant the state graph must be unique on state space; the asynchronous DMA controller specification is unsatisfied while state based was applied. From the state graph, if we try to insert an additional signal, the complication level is increased and suffers the state explosion.

Secondly, structural encoding technique is applied. The asynchronous DMA controller can be synthesized with this technique, but it shows some complexity in signal removal phase. Finally, template based is introduced. We generate the template STG from original STG using Petri-net level then we trace all non-input signals to small state space and derive it to Karnaugh map, if it shows conflicts, we have to insert some signals from template STG repeatedly, and we can get the final STG called STG with appropriate insertion point.

The final part of this paper is concluded with the complexity comparison between our template based technique, state based technique and structural encoding technique.

Manuscript received March 22, 2008 revised April 24, 2008. This work was supported in part by TJTTP-OECF (Thailand - Japan Technology Transfer Project – Japanese Overseas Economic Cooperation Fund).

Sufian Sudeng is with the Digital System Engineering Laboratory (DSEL), Department of Computer Engineering, Chulalongkorn University, Thailand (E-mail: sovanyy@msn.com).
Arthit Thongtak is with the Department of Computer Engineering, Chulaongkorn University, Thailand ((E-mail: Arthit@cp.eng.chula.ac.th).

II. Petri-Net and Signal Transition Graph

To be able to introduce the methods of synthesis of asynchronous circuits in subsequent sections, we will need a more formal definition of an STG. STGs are a particular type of labeled Petri-Nets, while the definition of Petri-net is a net is a triple $N_{df} = (P; T; F)$ such that P and T are disjoint sets of respectively places and transitions, and $F \subseteq (P \times T)$ [$(T \times P)$ is a flow relation. A marking of N is a multi set M of places, i.e., M : P -> {0, 1, 2…}. The transitions are associated with the changes in the values of binary variables. These variables can be associated with wires, when modeling interfaces between blocks, or with input, output and internal signals in a control circuit [5].

Signal Transition Graph (STG) is a quadruple $\Gamma = (N;M_0;Z;\lambda)$, where

- $\sum = (N;M_0)$ is a Petri net (PN) based on a net N = (P; T; F)
- $Z^{\pm}$ is a finite set of binary signals, which generates a finite alphabet $Z^{\pm} = Z^{\pm} \times \{+,-\}$ of signal transitions
- λ: T -> $Z^{\pm}$ is a labeling function.

An STG inherits the basic operation semantics from the behaviour underlying Petri-Net. In particular, this includes : the rules for transition enabling and firing, the notions of reachable markings, traces, and the temporal relation between transitions. STGs also inherit the various structural and behavioural properties and the corresponding classification of Petri-Nets Namely:

Choice place: A place is called a choice or conflict place if it has more than one output transition.

Marked graph and State machine: A Petri-Net is called a marked graph if each place has exactly one input and one output transition. A Petri-Net is called a state machine if each transition has exactly one input and one output place. Marked graph has no choice. Save state machine has no concurrency.

Free-choice: A choice place is called free-choice if every its output transition has only one input place. A Petri-Net is free-choice if all its choice places are free-choice.

Persistency: A transition *t* E *T* is called non-persistence if some reachable marking enables *t* together with another transition *t'*. Non-persistency of t with respect to t' is also called a direct conflict between t and t'. A Petri-Net is persistence if it does not contain any non-persistence transition.

Boundless and safeness: A Petri-Net k-bounded if for every reachable marking the number of tokens in any place is not greater than k (a place is called k-bounded if for every reachable marking the number of tokens in it is not greater than k). A Petri-Net is bounded, if there is a finite k for which it is k-bounded. A Petri-Net is safe if it is 1-bounded (1 bounded place is called a safe place).

Liveness: A Petri-Net is live if for every transition t and every reachable marking M there is a firing sequence that leads to a marking M' enabling t.

The properties for synthesizable STGs :

-**Persistency**–Excited signal fire, namely they cannot disable by other transition.

-**Consistency**–Signal strictly alternate between + and - .

-**Complete state coding** – Different marking must represent different states.

III. Asynchronous DMA Controller Specification

The Asynchronous processor is a master device on the asynchronous system bus [7], that it is able to initiate read and write transfers to all devices on the bus. The other peripherals are slave devices, which they are only able to response to read or write requests. The DMA controller is required to initiate read and write transfers to memory and also other I/O on the bus, DMA controller also required to be a master device.

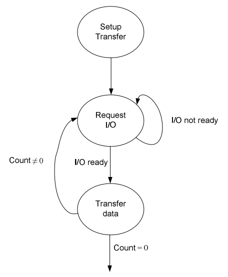

Figure 1: DMA basic

As there will be more than one master device in the system, asynchronous system bus specification required the presence of a bus arbiter. The bus arbiter selects which master is to have right to the bus at any one instant in time. An arbiter also exits for the asynchronous processor. Initially the DMA controller is programmed by asynchronous processor. This information is stored in internal DMA registers. Include following information: base address for transfer source, base address in memory to where the data is transferred and size of data transfer. The DMA controller also be slave on the asynchronous system bus in order the asynchronous processor access these registers.

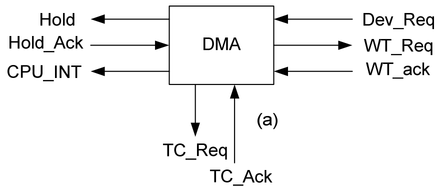

Figure 2: Asynchronous DMA Controller specification

The DMA controller waits for *Dev_req* line to assert and take over the asynchronous system bus from asynchronous processor as shown in figure 2 and 3. It checks its internal registers to obtained details of the transfer. The DMA controller read data from the source, and then writes it out to the memory, until the transfer is complete.

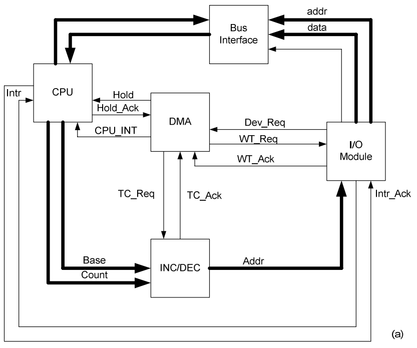

Figure 3: Asynchronous DMA controller architecture

The internal architecture is very closely based on the classic DMA controller. The architecture proposed of the DMA controller split into above functional units. The most complex of these units is timing and control unit, which consist of a large and complicated STG which described in the diagram on the figure 4.

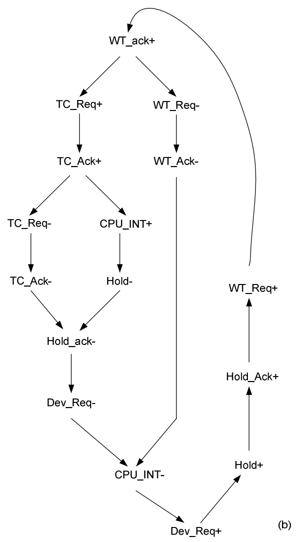

Figure 4: Signal transition graph

The DMA controller works follow on STG as shown in figure 4. Finally, DMA releases the usage of asynchronous system bus and activate the *CPU_INT* line in order to indicate the asynchronous processor that transfer was complete.

IV. Synthesis Using State Based Method

The purpose of this section is to present the synthesis of DMA controller using state-based technique. The key steps in this method are the generation of a state graph, which is a binary encoded reach ability graph of the underlying Petri-net, and deriving boolean equations for the output signals via their next state functions obtained from the state-graph.

The state graph shown in figure 5 is a state space of DMA controller specification that was derived from figure 4, the shadowed states are the conflict states, it clearly suffers to state explosion problem of state based method, to solve the explosion problem, an additional states are inserted, from state graph, it hardly finds an insertion of the internal signals due to many of redundant states, it's the major problem of state based technique.

State encoding, at this point, we must have noticed a peculiar situation for the value of the next-state function for signal that in two states with the same binary encoding. This binary encoding is assigned to the shadowed states in figure 5. The binary encoding of the state graph signals alone cannot determine the future behavior. Hence, an ambiguity arises when trying to define the next-state function. This ambiguity is illustrated in the Karnaugh map. When this occurs, the system is said to violate the Complete State Coding (CSC) property. Enforcing CSC is one of the most difficult problems in the synthesis of asynchronous circuits

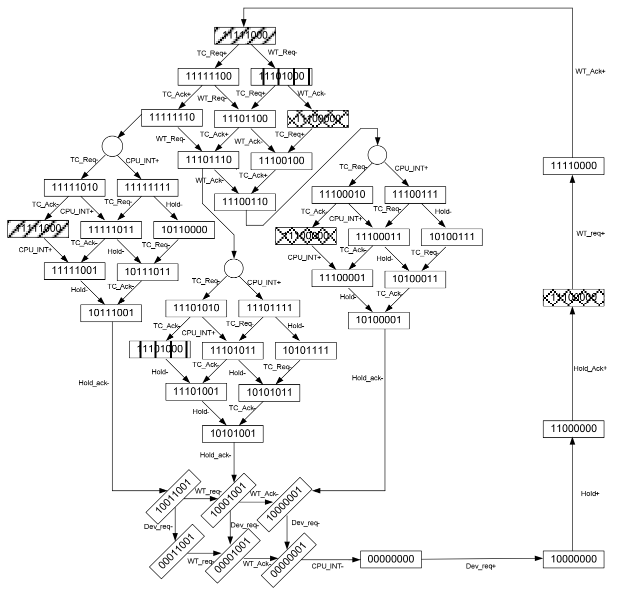

Figure 5: State graph

. This section shows a limitation and problem of state based synthesis. However, the solution of state based design is also possible; with inserting internal signals to solve the two conflicting states are disambiguated by the value of CSC, which is the last value in the binary vectors. Now Boolean minimization can be performed and logic equations can be obtained.

V. Synthesis Using Structural Encoding Method

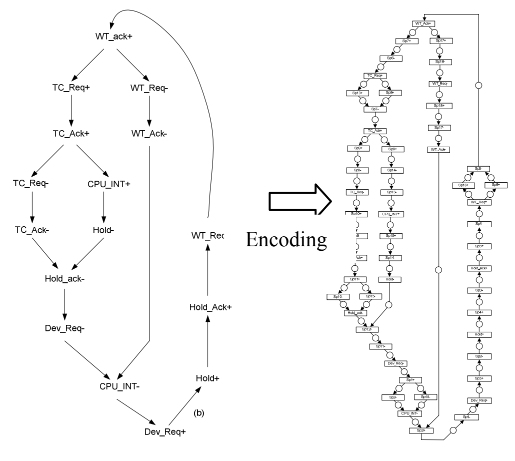

Figure 6: Encoded STG

The structural encoding scheme provides a way to avoid the state space explosion problem. The main benefit of using structural method is the ability to deal with large highly concurrent specifications, which cannot be tracked by state base method, the structural method for synthesis of asynchronous circuits is the class of mark graph. By transform the STG specification to Petri net, and then encode it to be well form specification as shown in figure 6 and 7.

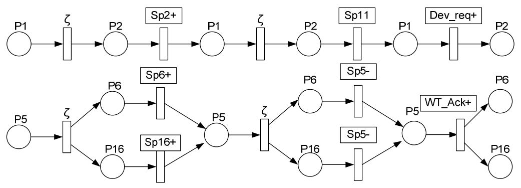

Figure 7: Encoding scheme (*Dev_Req+,WT_Ack+*)

The main idea of structural method is insertion of new signals in the initial specification in a way that unique encoding is guaranteed in the transform specification.

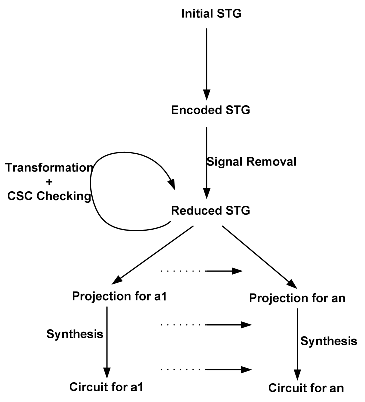

Figure 8: Structural encoding synthesis

Structural encoding technique, working at level of Petri-net can synthesize a big size of specification. State based is used in the final state of this method, when specification has been decomposing to smaller ones [5].

The synthesis flow is given a consistent STG, encode for all signals resulting an STG contains a new set of signals that ensure unique state and complete state coding property. Depending encoding technique applied, since many of encoding signal may unnecessary to guarantee unique and complete state coding, they are iteratively removed each signal using greedy heuristics until no more signal can be removed without violating unique and complete state coding property. The reduced STG next projected onto different sets of signals to implement each individual output signal. One the reduced STG is reached, it must be computed the complete state support for each non-input signal, applying CSC support algorithm as shown in next section. Afterward the projection of the STG into the CSC support for each non-input signal is performed, finally speed independent synthesis of each projection is performed, provide the small size of the projection, state based techniques can be applied for performing the synthesis. When synthesizing the projection for each non-input signal a, every signal is the projection but a is considered as an input signal. These prevent the logic synthesis to try to solve possible conflicts for the rest of signals.

VI. Synthesis Using Template Based Method

Structural encoding can be synthesizing a big size of specification; state based is used in the final state of this method, when specification has been decomposed to smaller ones. To design with structural encoding method, Petri-net level is applied, begins with encode original STG with two silent transition, the encoding element composes of two transitions and one place, then iterative signals reduction immediately with complete state checking, finally, STG with remaining signals are produced, and project for all non-inputs signal to get the small size CSC support, apply state based method for each CSC support to get a circuits for each non-input signals, However, structural based still consumed more wasting time in the design phase, especially signal reduction phase, each time of reduction, it must verify the persistence, consistence and complete state coding properties before reached the final reduction.

This work focuses on complexity reduction; our process begins with an encoding STG using Petri-net level in order to form a template STG. Then the projection to each non-input signals from an original STG is done. After that, we trace the projection to smaller state space. If the small state space shows conflicts, we have to insert balance signals from template STG. Unbalance signals are inserted after in case the space still shows conflicts. Finally, we can get the STG with appropriate insertion points which are used to be projected for CSC support on each non-input signal as shown in figure 9.

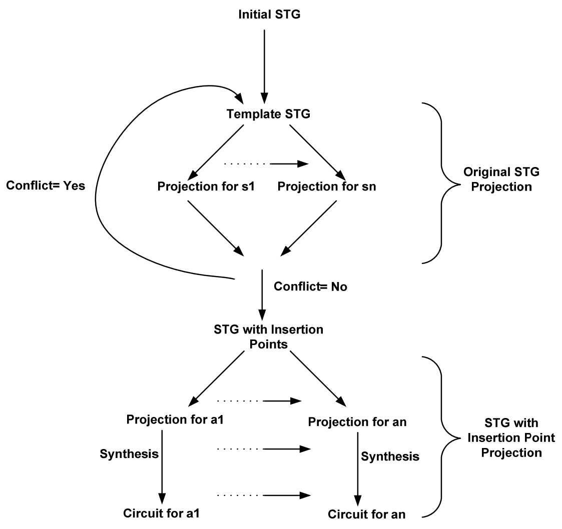

Figure 9: Template based synthesis

<u>*Template based Method (Insertion point calculation)*</u>

```
Let:
    a  = non input signal
    b  = balance signal
    ub = unbalance signal

Original STG Projection (a)
 While not Conflict do
```

```
        Trace a;
            If Conflict Then
                    Insert b from template STG;
    Else If Still Conflict Then
                    Insert ub from template STG;

    Else
      Trace a;
    End If;
```

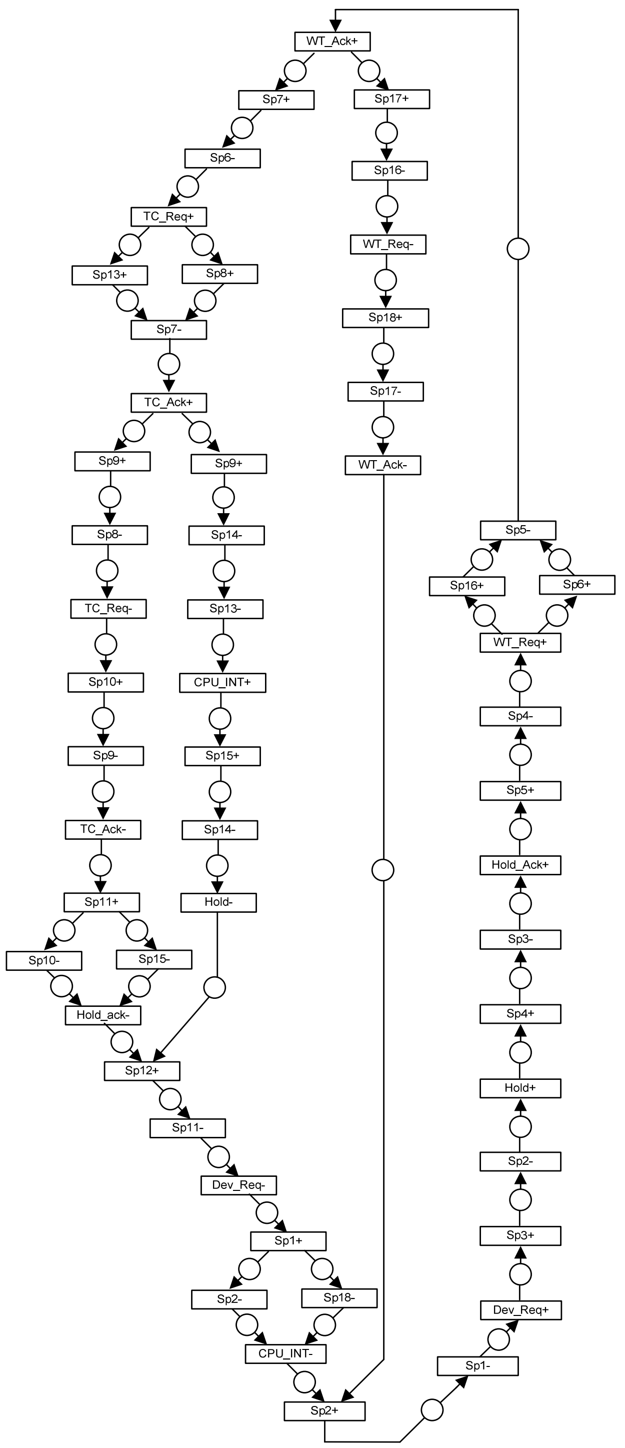

Figure 10: Template STG

Figure 10 shows the template STG that encoded with the encoding scheme in figure 7.

After the template based technique was applied. We can get the STG with appropriate insertion signals as shown in figure 11.

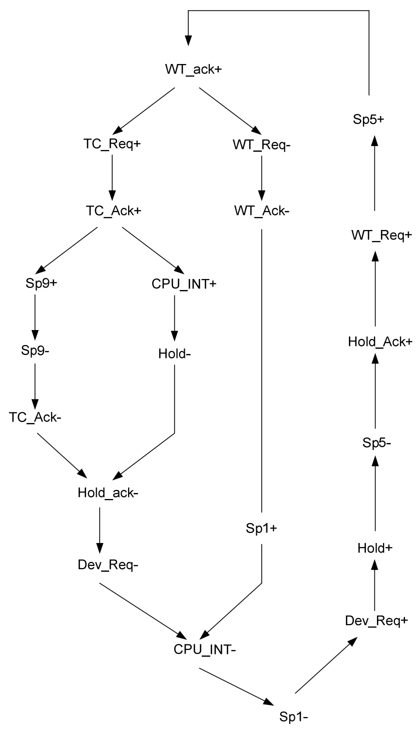

Figure 11: STG with appropriate insertion signals

CSC Support calculation[8]:

CSC Support (STG S, a) return CSC support of a
CSC support a = Trigger (a) U {a}
While it's still conflict do
Let b an unbalance signal in z
CSC Support for a = CSC Support for a U {b}

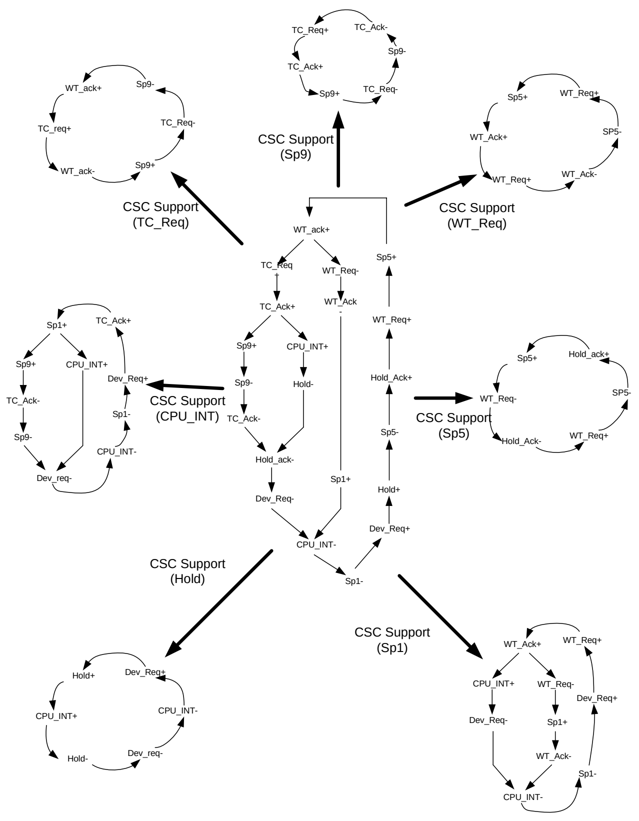

Figure 12: STG with Insertion Point and CSC support calculation

CSC support calculation for each non-input (output and internal signals) signal is calculated. Finally we got the small state graph which guarantee that it save from state explosion and also satisfied the synthesizing properties.

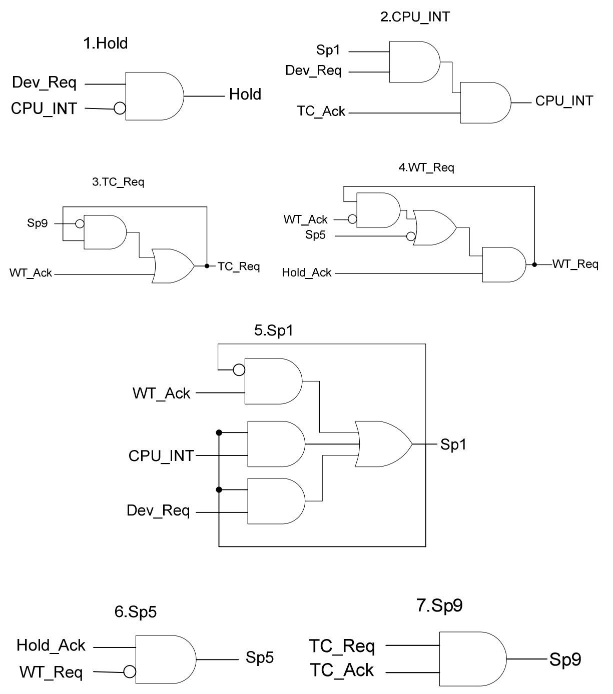

Figure 13: synthesized circuits from CSC support of Fig.12

VII. Result

This section presents complexity comparison between structural encoding method and template based method. The conventional state based method is not shown on a comparison table, it clearly suffers its exponential complexity of conflict state with the size of specification, and others difficult problem. Due to absence of synthesis tools for asynchronous controller synthesis, all circuits in table I were manually synthesis.

Circuits	Sign als	Structural Encoding		Template based	
		Iterative	Inserted signal	Iterative	Inserted signal
VME bus	6	**10**	2	**6**	2
Asyn DMA	8	**10**	6	**7**	6
Pipeline-Sched uler	14	**26**	2	**18**	2
QDI- bus	16	**28**	4	**20**	3
Pipeline control Block	13	**23**	3	**18**	3
Classical DMA	11	**13**	9	**8**	6
PPArbCSC (2,3)	10	**18**	2	**8**	2

Table 1: Result

VIII. Concluding Remarks

This paper proposes template based technique for reducing additional signals in signal transition graph (STG) based logic synthesis, and solves the reduction complexity of structural encoding method; our method is explained by asynchronous DMA controller specification. The final section showed the comparison between our template based method with state based and structural encoding method. The number of iterative signal removal according to our method is less than others, Roughly speaking, structural encoding is better than classical state based method, and template based method is less complexity than structural encoding.

Acknowledgement

The research is supported by TJTTP-OECF (Thailand - Japan Technology Transfer Project – Japanese Overseas Economic Cooperation Fund)

Reference

[1] S.B. Park. Synthesis of Asynchronous VLSI circuit from Signal Transition Graph Specifications, Phd. Thesis, Tokyo Institute of Technology, 1996.
[2] J. Carmona, J. Cortadella. ILP models for the synthesis of asynchronous control circuits. In Proc. International Conf. Computer-Aided Design (ICCAD), pp 818-825, November 2003.
[3] J. Carmona and J. Cortadella. State encoding of large asynchronous controllers. In Proc. ACM/IEEE Design Automation Conference, pp 939-944, July 2006.
[4] S.Sudeng, A.Thongtak: Asynchronous System Bus Enhancement by Interrupt and DMA Technique International Annual Conference (ECTI-CON2007), Mae Fah Luang University, Thailand on May 9-12, 2007.
[5] J.Carmona, J. Cortadella, and E. Pastor. A structural encoding technique for the synthesis of asynchronous circuits. In Int. Conf. on Application of Concurrency to System Design,pages 157-166, June 2001.
[6] T.-A. Chu, Synthesis of self-timed VLSI circuits from graph-theoretic specifications, Ph.D. dissertation, MIT Lab. Comput. Sci., Cambridge, MA, Jun. 1987.
[7] S.Sudeng and A.Thongtak FPGA Implementation of Quasi-Delay Insensitive Microprocessor. The World Congress on Engineering and Computer Science (WCECS2007), Clark Kerr Campus, University of California Berkeley, San Francisco, USA on 24-26 October 2007.
[8] S.Sudeng and A.Thongtak "Signal Transition Graph Based Logic Synthesis for Asynchronous Control Circuits Using Template Based Method" IEEE Tencon 2007, IEEE Region 10 conference (IEEE Tencon2007), Taipei International Convention center, Taipei, Taiwan on Oct 30- Nov 2, 2007.