Ata 14 de reunião do projeto TETHA - 08/Junho/2006

Presentes (Equipe PUC):

Nome	e-mail	Dedicação
Fernando Moraes	moraes@inf.pucrs.br	coordenador
Ney Laert Vilar Calazans	calazans@inf.pucrs.br	coordenador
Everton Carara	carara@inf.pucrs.br	40
Erico Bastos	ebastos@inf.pucrs.br	40
Alzemiro Henrique Lucas	alzemiro@gmail.com	30
Guilherme Montez Guindani	ac107029@inf.pucrs.br	30
Samuel dos Santos Marczak	samuelmarczak@yahoo.com.br	30
Taciano Ares Rodolfo	taciano@inf.pucrs.br	30

Pauta da reunião:

1. **Guindani e Taciano.**

   - Dificuldades na compreensão da interface Wishbone com o bloco de controle do MAC;
   - O Ney irá auxiliar o grupo nesta tarefa.

2. **Alzemiro.**

   - Test bench para ler inúmeros pacotes. **Não haverá** geração de *padding*. O *offset* da célula indica o último byte válido;
   - Implementar a separação de células conforme a sessão, e transmitir para arquivos de saída.

3. **Érico.**

   - Deve validar as tabelas de rotemento MAC;
   - O procedimento para envio à rede é como segue: (*i*) a primeira célula de um pacote contém o destino do pacote, o NI busca na tabela de roteamento a relação endereço MAC – endereço NoC e armazena em um registrador interno este endereço NoC, utilizando-o para todo o pacote; (*ii*) as demais células utilizam o mesmo destino. Haverá um estabelecimento de circuito para cada célula do pacote. **Não** há ainda nesta fase endereço inexistente na tabela, pois isto implicará em ***broadcast***;
   - Deve ser definido o algoritmo de recepção de células para a escrita na tabela de roteamento. Observar que neste caso deve-se atentar ao controle de sessão, pois o NI poderá estar recebendo células de diferentes origens. O procedimento básico é o seguinte: ao chegar a primeira célula de uma sessão, escreve-se o endereço MAC e endereço origem na tabela de roteamento. **Política de substituição atual**: inicialmente a tabela conterá três a quatro endereços válidos, e novos endereços serão acrescidos de forma circular;
   - Documentar os algoritmos.

4. **Samuel.**

   - Responsável pela integração do sistema, *scripts* modelsim e validação da plataforma MOTIMT.

5. **Carara.**

   - Irá implementar o controle de sessão nos reteadores, a fim de permitir que cada roteador possa receber células de até '*n*' pacotes simultaneamente, onde '*n*' deve ser parametrizável. Inicialmente '*n*' será igual a 4;
   - Adição de sinais *sideband* entre o roteador e a interface de rede: número da sessão e *ack/nack* para sinalizar aceite ou não de estabelecimento de circuito para uma dada célula. Se o sinal *nack* for ativado, a origem deverá esperar aguardar '*k*' ciclos de clock antes de retransmitir, onde 'k' deve ser parametrizável. Inicialmente '*k*' será igual a 200.

6. **Modificação na arquitetura MOTIMT.**

   - Foram acrescentados dois sinais: off_set_ni (off_set_pc) e start_packet_ni (start_packet_pc) figura 1;

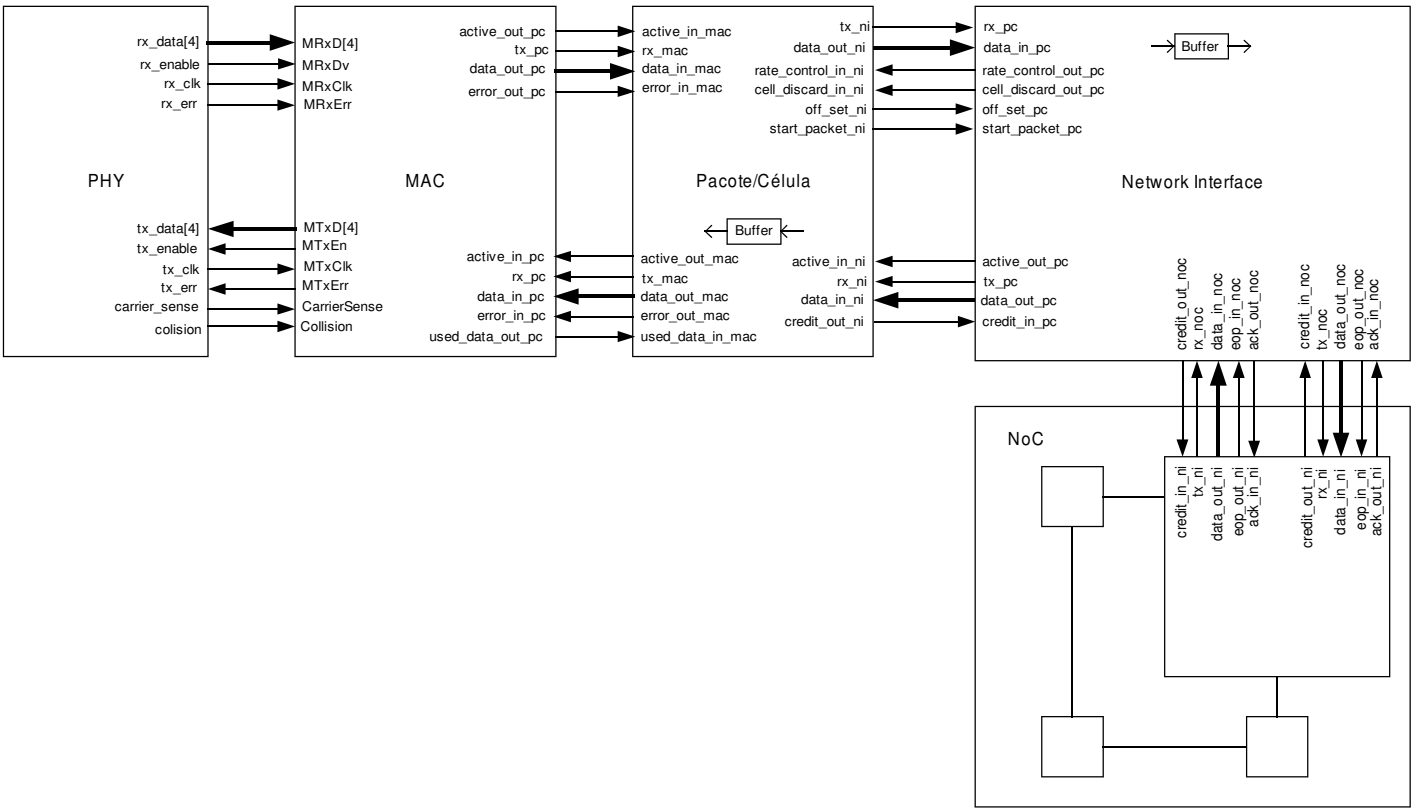


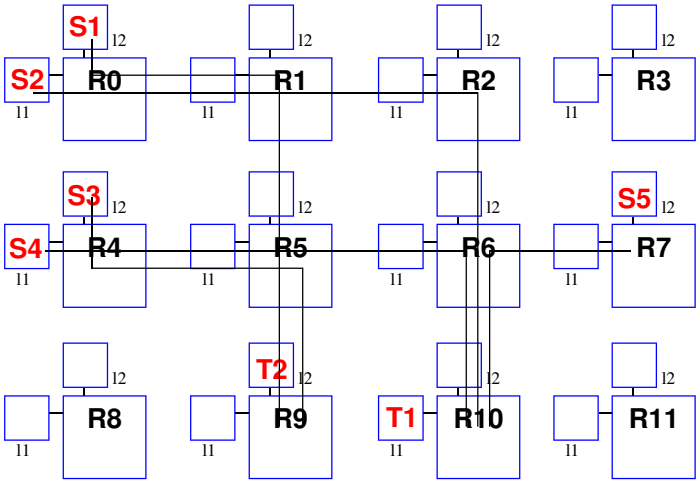
Figura 1: Arquitetura MOTIMT.

Próximos Trabalhos:

- Uso do CVS. Atrasado devido a problemas técnicos de acesso ao servidor;
- Apresentar a uma simulação válida para a reunião do dia **20/Junho**.

Apresento abaixo (Moraes) os cenários iniciais de teste, que deveremos apresentar no dia 20:

Cenário 1: Teste de sessão, múltiplas portas de origem, aprendizagem, uso dos dois canais físicos e saturação.



- Tabelas de roteamento fixas no início para S1, S2, S3, S4, S5
- Tabelas de roteamento vazias para T1, T2

- Padrões de comunicação (S➔source e T➔target):
    - S1➔ T2
    - S3 ➔ T2
    - S2➔ T1
    - S4 ➔ T1
    - S5 ➔ T1

- Configuração dos geradores de teste: todos com 100 pacotes de 1024 bytes, com intervalo de tempo mínimo (mas configurável) entre os pacotes.
- Estaremos nesta simulação com apenas 500 pacotes (500kB) de dados.

Cenário 1: Teste de sessão, múltiplas portas de origem, aprendizagem, uso dos dois canais físicos e saturação.

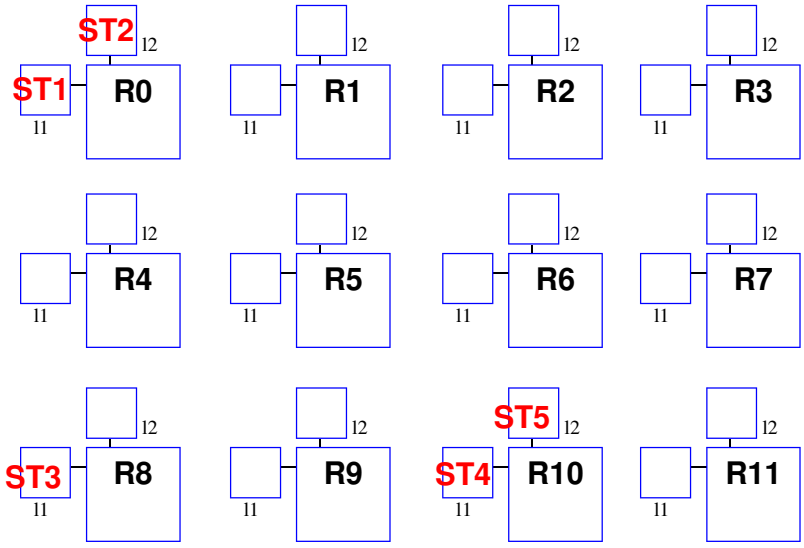


Diagrama temporal de envio de pacotes de 1024 bytes:

- Inicialmente apenas os roteadores R8 e R10 tem dados na tabela de roteamento, apontando para o reteador zero.
- No início da simulação os roteadores R8 e R10 enviam pacotes para o roteador R0, fazendo com que haja aprendizado em R0
- O test_bench do roteador R0 deve ser capaz de iniciar o envio de pacotes após um *tempo x*, que deve fazer parte do arquivo de pacotes. Após o tempo *x* cada porta local do roteador R0 envia pacotes para os destinos ST3, ST4 e ST5 simultaneamente, para avaliarmos o comportamento do controle de sessão e canais físicos.