4 Memory Architecture

4.1 Memory Address Map

The Epiphany architecture uses a single, flat address space consisting of 2^{32} 8-bit bytes. Byte addresses are treated as unsigned numbers, running from 0 to $2^{32} - 1$. This address space is regarded as consisting of 2^{30} 32-bit words, each of whose addresses is word-aligned, which means that the address is divisible by 4. The word whose word-aligned address is A consists of the four bytes with addresses A, A+1, A+2 and A+3. Each mesh node has a local, aliased, range of memory that is accessible by the mesh node itself starting at address 0x0 and ending at address 0x00007FFF. Each mesh node also has a globally addressable ID that allows communication with all other mesh nodes in the system. The mesh-node ID consists of 6 row-ID bits and 6 column-ID bits situated at the upper most-significant bits (MSBs) of the address space. The complete memory map for the 32 bit Epiphany architecture is shown in Figure 5.

Figure 5: Epiphany Global Address Map

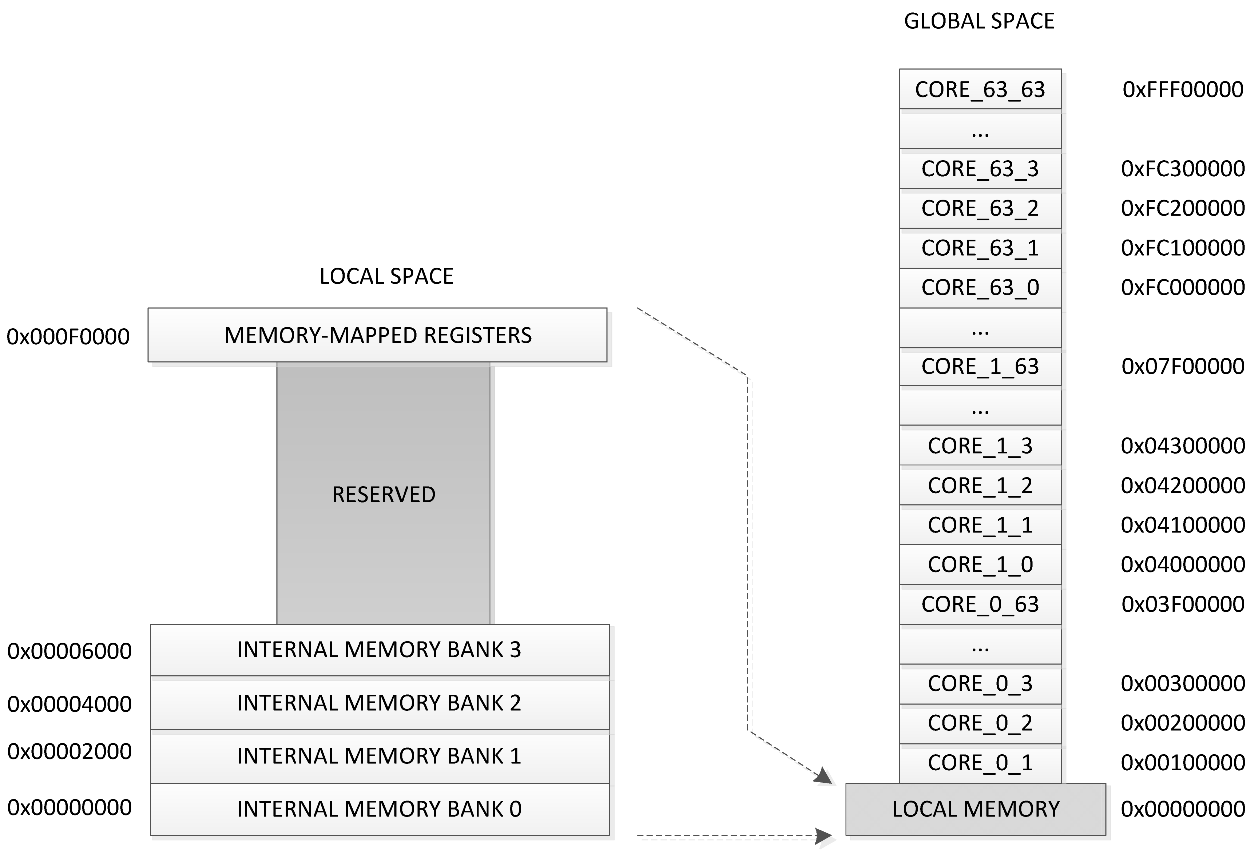
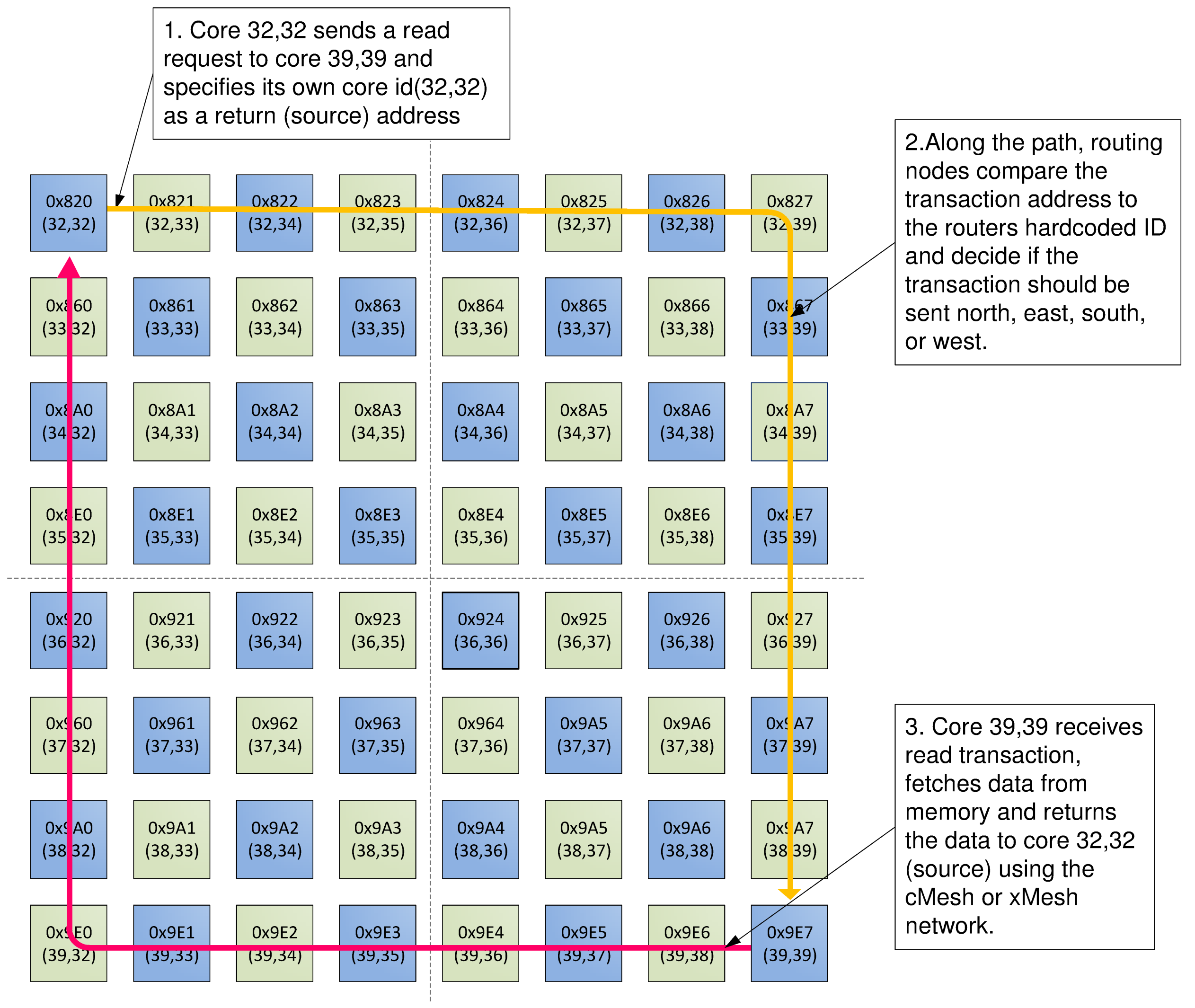


Figure 8: eMesh™ Routing Example



5.4 Direct Inter-Core Communication

Figure 9 shows how the shared-memory architecture and the eMesh network work productively together. In the example, a dot-product routine writes its result to a memory location in another mesh node. The only thing required to pass data from one node to another is the setting of a pointer. The hardware decodes the transaction and determines whether it belongs to the local node's memory or to another node's memory. Since the on-chip cMesh network can accept write transactions at the same rate that a processor core can dispatch them, the example runs without pipeline stalls, despite executing a node-to-node write in the middle of the program stream. Using this method, programmers can reduce the cost of write-based inter-node communication to zero.

Figure 9: Pointer Manipulation Example

C-CODE		ASSEMBLY
//VecA array at 0x82002000		//R0=pointer to VecA
//VecB array at 0x82004000		//R2=pointer to VecB
//remote_res at 0x92004000		//R6=pointer to remote_res
		//R4=loc_sum;
for (i=0; i<100; i++){		MOV R5,#100;
loc_sum+=vecA[i]*vecB[i];	→	_L: LDR R1,[R0],#1;
}		LDR R3,[R2],#1;
remote_res=loc_sum;		FMADD R4,R1,R3;
		SUB R5,R5,#1;
		BNE _L;
		STR R4,[R6];

5.5 Arbitration Scheme

The routers at every node in all three mesh networks contain round-robin arbiters. The arbitration hardware, in combination with the routing topologies, ensures that there are no deadlocks. The round-robin scheme also ensures that there is some split of available bandwidth between the competing agents on the network. The large on-chip bandwidth and non-blocking nature of the write network guarantees that no agent needs to wait more than a few clock cycles for access to the mesh. Applications requiring exact and deterministic bandwidth can implement network-resource interleaving in software.

5.6 Data Sizes and Alignment

The eMesh network supports byte, halfword, word, or doubleword atomic transactions. Mesh data is always aligned to the least-significant bits (LSBs). Maximum bandwidth is obtained with doubleword transactions. All transactions should have addresses aligned according to the transaction data size.