Computing Surface

Communications Network Overview

S1002–10M105.05

meiko

The information supplied in this document is believed to be true but no liability is assumed for its use or for the infringements of the rights of others resulting from its use. No licence or other rights are granted in respect of any rights owned by any of the organisations mentioned herein.

The specifications listed in this document are subject to change without notice.

Meiko, CS-2, Computing Surface, and CSTools are trademarks of Meiko Limited. Sun, Sun and a numeric suffix, Solaris, SunOS, AnswerBook, NFS, XView, and OpenWindows are trademarks of Sun Microsystems, Inc. All SPARC trademarks are trademarks or registered trademarks of SPARC International, Inc. Unix, Unix System V, and OpenLook are registered trademarks of Unix System Laboratories, Inc. The X Windows System is a trademark of the Massachusetts Institute of Technology. AVS is a trademark of Advanced Visual Systems Inc. Verilog is a registered trademark of Cadence Design Systems, Inc. All other trademarks are acknowledged.

Meiko's address in the US is:

Meiko
130 Baker Avenue
Concord MA01742

508 371 0088
Fax: 508 371 7516

Meiko's full address in the UK is:

Meiko Limited
650 Aztec West
Bristol
BS12 4SD

Tel: 01454 616171
Fax: 01454 618188

Issue Status: Draft □
Preliminary □
Release ☒
Obsolete □

Circulation Control: *External*

Contents

GeneralDescription 1

Effective cooperation between processing elements (PEs) is a crucial factor in determining the overall sustained performance of a Massively Parallel Processing (MPP) system.

In designing the CS-2 architecture, Meiko has concentrated on minimizing the impact of sharing work between processors. The effect of this is to increase the number of processors that can be effectively used to solve a problem, improving the performance of existing parallel programs, and making parallel processing effective for a significantly wider range of applications.

Every processing element in a CS-2 system has its own, dedicated interface to the communications network: a Meiko designed communications processor. The communications processor has a SPARC shared memory interface and two data links, these links connect the communications processors to Meiko designed cross-point switches.

This document provides an overview of the design of the communications network. For more information about the architecture of the communications processor see the *Communications Processor* Overview.

Network Characteristics

The design of the CS-2 data network builds on Meiko's considerable expertise in the field of MPP systems. From the outset the communications network was designed with several key characteristics in mind:

- Full connectivity.
- Low latency.
- High Bandwidth.
- Fault tolerance.
- Deadlock freedom.
- Scalability.

Full Connectivity

Every processing element (PE) has the ability to access memory on any other PE. Messages pass from the source to destination PEs via a dynamically switched network of active switch components. The network is fully connected, allowing a machine with *n* PEs to sustain *n* simultaneous transfers between arbitrarily selected pairs of PEs at full bandwidth.

The communication network does not use the PEs as part of the network, only as gateways on to it. This ensures that node resources (such as CPU and memory bandwidth) are not affected by unrelated network traffic.

Low Latency

Inter-process communications latency has two components, start-up latency (which is covered in the *Communications Processor Overview*) and network latency. The CS-2 communication network is designed to minimize and hide network latency. Wormhole routing is used to reduce the latency through each switch stage, and the overall network topology is designed to minimize the number of stages through which a message passes. The low level communication protocols allow overlapped message acknowledgments, and the message packet size is dynamically adjusted so that it is always sufficient for full overlapping to occur.

CS-2 communications start-up latency are less than 10µs, network latencies are less than 200ηs per switch.

High Bandwidth

The communication bandwidth in an MPP system should be chosen to give an appropriate compute communications ratio for current PE technology. The network design should ensure that additional bandwidth can be added to maintain the compute/communication ratio as the performance of the PEs improves with time. Although the actual required compute/communications ratio is application specific, the higher the network bandwidth the more generally applicable the MPP system will be.

CS-2 data links are byte wide in each direction and operate at 70 MHz. Usable bandwidth (after protocol overheads) is 50 Mbytes/s/link in each direction. Bisectional bandwidth of the CS-2 network increases linearly with the number of PEs. A 1024 PE machine has a bisectional bandwidth of over 50 Gbytes/s.

Fault Tolerance

The network for a very large MPP system will of necessity consist of a very large number of components. Moreover for large systems a significant number of cables and connectors will be required. Under these circumstances reliability becomes a major issue. Tolerance to occasional failures by the provision of multiple routes through the network is desirable for small systems, and essential for very large systems.

CS-2 systems have two fully independent network layers and each PE is connected to both layers. In addition each layer provides multiple routes between each arbitrarily selected pair of PEs. The hardware link protocol uses Cyclic Redundancy Checks (CRCs) to detect errors on each link; failed transmissions are not committed to memory, but cause the data to be resent. All network errors are flagged to the System Administrator; permanently defective links can be removed from service.

Deadlock Freedom

Routing through multistage networks is essentially a dynamic resource allocation problem and, because multiple PEs are attempting to acquire sets of route hops simultaneously, there is the potential for deadlock. The most common deadlock avoidance strategy is always to allocate resources in a fixed order. With wormhole routing, since the resources are allocated as the message wormholes through a network, this affects routing strategy for a given topology. For example in a hypercube or a grid, deadlock free routing is possible by ensuring that a PE routes by resolving the address one dimension at a time in ascending order. Note: that this actually removes the fault tolerance of the network; between PEs that differ by more than one dimension there are many possible routes, but only one can be used without risk of deadlock.

Scalability

The requirement for scalability within a network is one of the most difficult to achieve in actual systems. The three factors that need to be considered are, growth in network latency with scaling, growth in network cost, and growth in bisectional bandwidth.

The scalability properties of various network topologies are:

Type	Number of Switches	Number of Links	Latency	Bisectional Bandwidth
Ring	N	N	$N-1$	2
d dimensional grid	N	dN	$d\sqrt[d]{N}$	$\sqrt[d]{N}$
Arity *d* Omega net	$N\log_d N$	$(dN\log_d N)/2$	$\log_d N$	N
Arity *d* benes net	$2N\log_d N$	$dN\log_d N$	$2\log_d N$	N
Crosspoint	N^2	N^2	1	N

Where N is the number of processors in the machine, *Number of Links* is the total number of connections between switches, *Latency* is the worst case number of switches which must be passed through, and *Bisectional Bandwidth* is the worst case bandwidth between two halves of the machine.

For scalability it is essential that the bisectional bandwidth of the machine increases linearly with the number of processors. This is necessary because many important problems cannot be parallelised without requiring long distance communication (for example, FFT, and matrix transposition).

The cost (both in switches and wires) of a full crosspoint switch increases as the square of the number of processors. Adoption of this network therefore leads to a machine in which switch and wire costs rapidly dominate when significant numbers of processors are used. For the logarithmic networks the switch and wire costs increase only logarithmically faster than the number of processors. It is therefore possible to build machines which contain significantly more processors before the switch costs dominate and the machine ceases to be cost effective.

The crosspoint has the advantages of contention freedom and constant network latency for all routes. However, although the worst case latency in a logarithmic network increases slowly with the number of processors, they can be arranged so as to ensure that this increase only occurs when long distance communication is required—performance is not dependent upon exploiting locality of reference, but doing so is beneficial.

The arity of the logarithmic network is the size of the crosspoint switch from which the network is built. So if the crosspoint is built from 2×2 switches it will have arity of 2. The choice of switch arity is highly influenced by the available packaging technology, since given a limited number of pins to connect into a switch there is a reciprocal relationship between the arity of the switch and the number of wires in each link. As the bandwidth of a link is directly related to the number of wires over which it is carried, this translates into a choice between a high arity switch which can switch many low bandwidth links, or a low arity switch for few high bandwidth links.

Logarithmic Networks

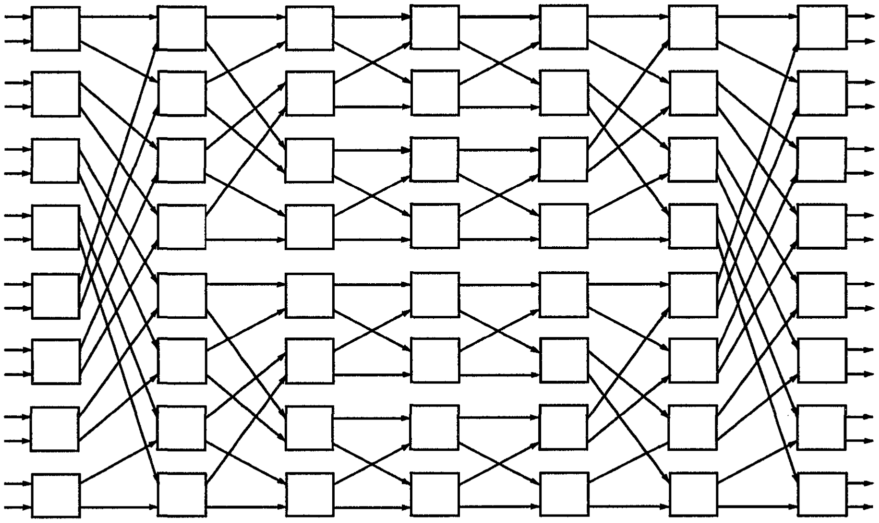
In order to analyze the CS-2 network it is useful to understand the characteristics of the Benes and Omega networks.

The main attraction of the Benes network is that it can be proved to have equivalent functionality to a full crosspoint (see Hockney and Jesshope[1] for a review)—any permutation of inputs can be connected to any permutation of outputs without contention. There are also multiple routes between any input-output pair. Calculating the routing to ensure that the routes are allocated without congestion for any given permutation is, however, a non-trivial problem.

[1] R.W.Hockney & C.R.Jesshope. Parallel Computers 2. Pub. Adam Hilger.

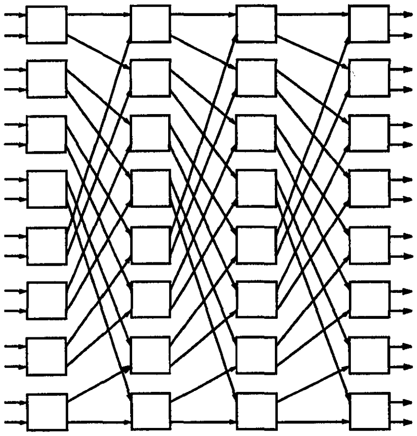
This problem has been solved for a number of interesting special cases communication patterns: rings, grids, hypercubes etc. There has also been extensive simulation of these networks under a wide variety of loadings.

Figure 1-1 16 Processor Benes Network



In an Omega network there is only one possible route for each input-output pair. Not all possible permutations are possible without blocking, although common geometric patterns such as shifts and FFT butterflies can be shown to be contention free.

Figure 1-2 16 Processor Omega Network



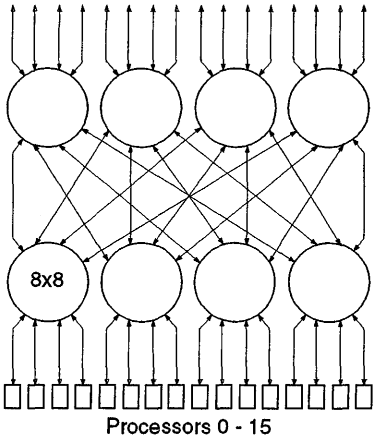
The CS–2 Communications Network 2

CS–2 uses a logarithmic network constructed from 8 way crosspoint switches (see Chapter 3 for details of their implementation) and bidirectional links.

For the purposes of this analysis it can be considered to be a Benes network folded about its centre line, with each switch chip rolling up the functionality of eight of the unidirectional two way switches.

Bandwidth is constant at each stage of the network, and there are as many links out (for expansion) as there are processors. Larger networks are constructed by taking four networks and connecting them with a higher stage of switches. A 16 processor network is illustrated in Figure 2-1.

Figure 2-1 One layer of a 2–stage CS –2 network. 16 processors are connected to stage 1, 16 links connect stage 1 to stage 2, and 16 links are available for expansion.



The scaling characteristics of the CS–2 network are shown in the table below; note that the latency is measured in switch stages for a route which has to go to the highest stage in the network.

Processors	Stages	Total Switches	Latency
4	1	1	1
16	2	8	3
64	3	48	5
256	4	256	7
1024	5	1280	9
4096	6	6168	11

One aspect of implementing the network using bidirectional switches is that routes which are relatively local do not need to go to the high stages of the switch hierarchy. So, for example, a communication to a PE which is in the same cluster of 16 processors only needs to pass through 3 switches irrespective of the total network size.

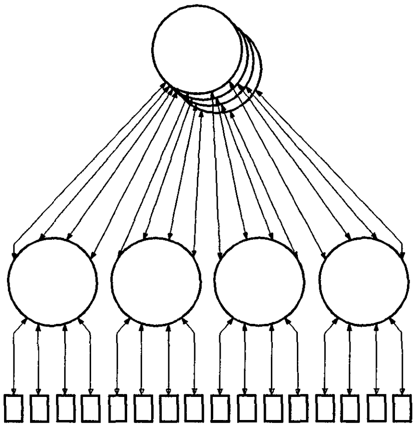
To broadcast to a range of outputs it is necessary to ascend the switch hierarchy to a point from which all the target PEs can be reached. From this point the broadcast then fans out to the target range of processors.

Comparison With Fat-Tree Networks

The multi-stage network used in the CS–2 machine can also be considered as a ``fat tree''. In Figure 2-1 we see that for each of the higher layer switches has identical connections to the lower stages. If this is simply redrawn as shown in Figure 2-2 we get the ``fat tree'' structure.

In fat trees packets do not always have to go to the top of the tree; packets are routed back down at the first node possible. This means that for problems which have locality of reference in communications, bandwidth at higher levels of the tree can be reduced. Exploiting the benefits of locality by reducing upper level network bandwidth has the effect of making process placement more significant. Although the CS–2 network permits this local packet routing, the bandwidth is not reduced in the higher level. This preserves the properties of Benes and Omega networks.

Figure 2-2 One layer of a 16 processor CS-2 network drawn as a fat tree.



Further properties of ``fat trees'' are described by Leiserson[1]

Characterising a CS-2 Network

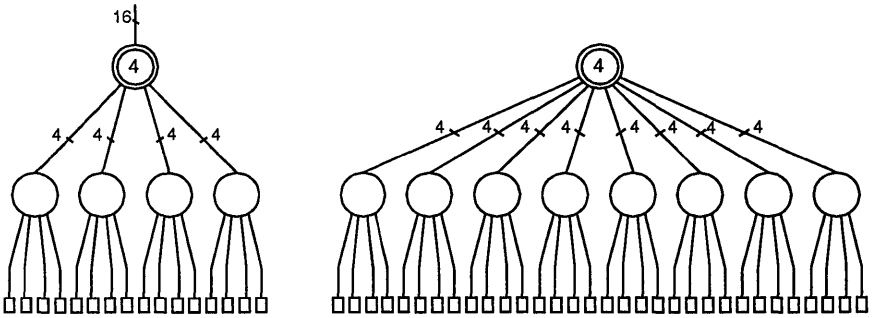
Logarithmic, or multi-stage, switch networks are described in a variety of ways by different people. The scheme used by Meiko is outlined below.

For a machine with *N* processors the size of its network is defined by one parameter: *size*. The position of a processing element is defined by two parameters: *level* and network *identifier.* The position of a switch in the network is defined by four parameters: *layer, level, network identifier,* and *plane.*

Every processor in a (complete) network is connected via a data link to a switch in the lowest stage, these switches are then connected to higher stages, etc and *N* links emerge from the top of the network. These links can be used to connect to further stages, or if we forgo the ability to expand they can be used to double the size of the network without introducing an extra stage (see Figure 2-3).

[1] C.E.Leiserson. Fat-Trees: Universal Networks for hardware-Efficient Supercomputing. IEEE Transactions on Computers, Volume C-34 number 10 (Oct. 1985). pp 892-901.

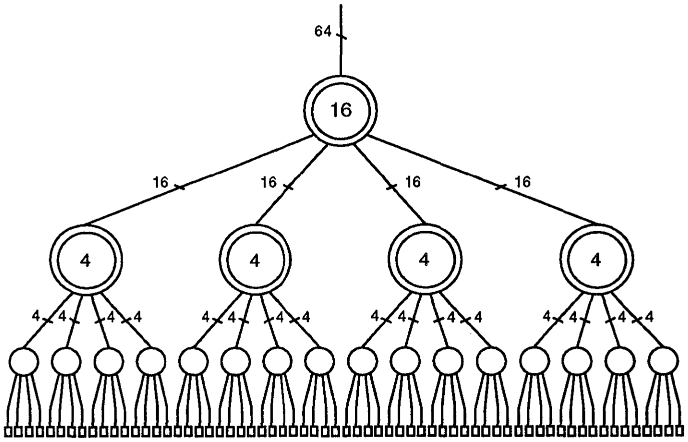
Figure 2-3 Doubling the size of a CS-2 network.



We use a binary form for network size, equal to the number of bits in the network identifier of the lowest processor in the network. This is used because the top stage of the network can use either 4 or 8 links.

A network has [*size*/2] stages, indexed by the parameter level. The top stage is 0. The deepest processors in the network have *level=size*. A network supports between $2^{(size-2)}+1$ and 2^{size} processors. Note: it is not necessary for the switch network to be complete. Figure 2-4 illustrates a network of size 6.

Figure 2-4 One layer of 64 processor (size 6) CS-2 network.



There are a variety of ways of drawing these networks (see the *CS–2 Product Description* for two other examples). To draw (or manufacture!) them without crossing data links you need one more dimension than there are stages in the network.

A CS–2 machine has 2 completely independent identical switch networks. These networks are indexed by the parameter layer. Processors are connected to both layers, switches are in one layer or the other.

The position of each processing element is uniquely determined by its network identifier and level, which describe the route to it from all points at the top of the network (*level*=0). Routes down are written <0-7>.<0-3>.<0-3> ... working down from the top of the network. Each digit represents the output link used on a network switch. For example, in Figure 2-4 processor 0 has route 0.0.0, and processor 17 has route 1.0.1. Note that the route is the same for all starting points at the top of the network. Network identifiers of communications processors (leaves of the network) are sometimes called Elan Identifiers.

Each stage of the switch network has $2^{(size-2)}$ switches, and 2^{level} distinct routes from the top of the network. The network identifier of a switch indexes the distinct routes within each level. Within each stage there are $2^{(size-level-2)}$ switches with the same route from the top of the network.

Network Implementation 3

The CS–2 communications network is constructed from a VLSI packet switch ASIC — the Elite Network Switch. Interfacing between the network and the processors is performed by a second device, the Elan Communications Processor. Switches are connected to each other and to communications processors by byte wide bidirectional links.

The Link Protocols

The choice of a byte wide link protocol is dictated by a number of factors. The link must be wide enough to meet the bandwidth requirements of the processor, but must not be so large that the number of I/O pins on the devices becomes prohibitively large. The implementation that Meiko selected uses 20 wires for each bidirectional link, 10 in each direction. When clocked at 70MHz this yields a bandwidth of 50Mbytes/s (after allowing for protocol overheads) in each direction. This level of performance and the underlying protocol format is appropriate for optic fibre communication over long distances (the link can be converted to a 630MHz data stream).

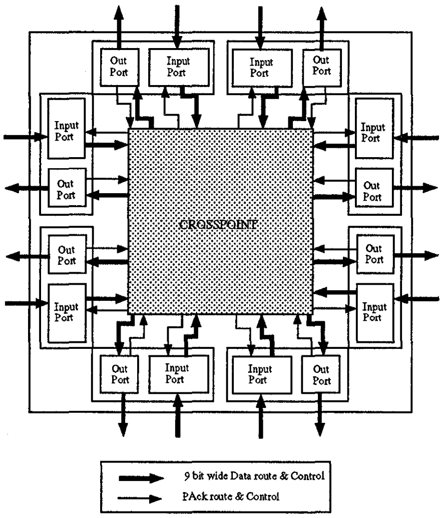
The use of bidirectional links permits flow control and acknowledge tokens to be multiplexed onto the return link. The low level flow control allows buffering of the data at the line level so that communications clock frequencies in excess of the round trip delay can be used. The interface is asynchronous and is tolerant to a 200ppm frequency difference between the ends. This means that each end can have its own clock, substantially simplifying construction of large systems.

The Meiko Elite Network Switch

The Elite switch is capable of switching eight independent links, each byte wide. The switch is a full crosspoint, allowing any permutation of inputs and outputs to be achieved without contention. For each data route through the switch a separate return route exists, ensuring that acknowledgements are never congested by data on the network.

The switch component contains a broadcast function that allows incoming data to be broadcast to any contiguous range of output links. The switch contains logic to recombine the acknowledge or not-acknowledge tokens from each of the broadcast destinations. To allow broadcasts to ranges of outputs over multiple switches the switch topology must be hierarchical.

Figure 3-1 Meiko Elite network switch.



The data passing through a switch is CRC checked at each switch. If a failure is detected the message is aborted, an error count is incremented, and the packet is negatively acknowledged. This ensures that incorrect data is removed from the network as soon as possible.

Routing within the switch is byte steered. On entry into a switch the first byte of any packet is interpreted as the destination output or range of outputs. This byte is stripped off within the switch so that the next byte is used for routing in the following switch. The latency through each switch device is 7 clock cycles for outgoing data, and 5 cycles for returning acknowledge tokens. The switch contains no routing tables of any sort. The translation between destination processor and route information is performed entirely on the communications processor, where it can be more easily modified or updated.

Although the switch component is an 8×8 crosspoint, the use of bidirectional links means that for the purposes of constructing logarithmic networks the effective radix is 4.

Each switch has a performance monitoring and diagnostic interface connected to the CS–2 control network. This allows collection of statistics on error rates and network loading.

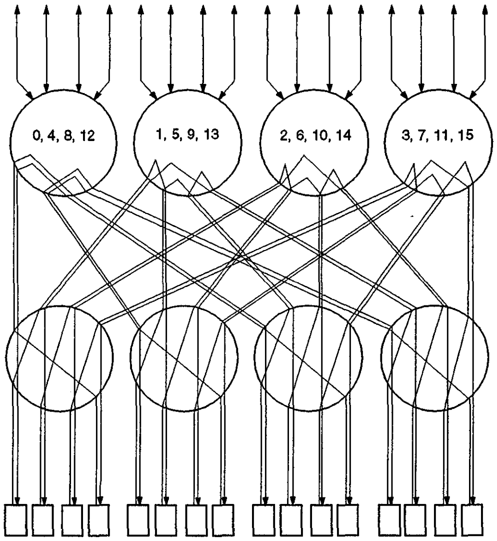
Routing Algorithms

Although the CS–2 data network can have the congestion properties of a full crosspoint, achieving this requires allocation of routes in a non-contending fashion. In the CS–2 network the route is predetermined by the communications processor. By storing the route information in the Elan it becomes easier to change the routing algorithm, due to machine reconfiguration or link failure for example.

The translation from a processor address to network route is handled in the communications processor by a look-up, the table is stored in the memory of the PE and indexed by destination processor. Each table entry contains four alternative routes to the destination processor, one of which is selected. The specification of alternative routes allows the even distribution of traffic throughout the network, although all four routes may be identical when this is undesirable. Each PE maintains its own look-up table which may be different to the others, thus enabling any function of source/destination addressing to be used from.

One simple routing function is to direct all data for the same destination processor through a single switch node at the top of the hierarchy. This allows the network to perform two functions: data distribution, and distributed arbitration for use where many senders wish to communicate with the same processor simultaneously. By adopting this strategy we ensure that if blocking does occur, it does so as soon as possible, and consumes little of the network resource. Using this simple algorithm has the effect of reducing the network to an Omega network — essentially the second, return part, of the network is guaranteed non blocking, and performs a simple data ordering operation. By virtue of its similarity to an Omega network, this network will be non-blocking for arbitrary shifts and FFT style permutations.

Figure 3-2 Shift by 5 on a 16 processor CS-2 network.



The programmable nature of the CS–2 communication network allows users (who are so inclined) to design their own routing algorithms. This permits optimisation of routing for specific traffic patterns or study of the effect of routing strategy on network performance.

Conclusions 4

The CS–2 network provides a flexible solution to the problem of connecting together large numbers of processing elements. The network can provide equivalent performance to a full crosspoint, but can be simplified where this level of interconnect is not required. The combination of Meiko Elan and Elite network technology allows considerable flexibility in the choice of routing algorithm.

The communications co-processor uses a lookup table to map abstract processor addresses to switch network routes. By maintaining the lookup tables within the PE memory they are easier to modify to reflect changing workload or network failures. By maintaining separate lookup tables on each communications processor, any function of address mapping may be implemented. The Elan communications processor acts as a gateway into the CS–2 switch network.

The Elite network switch is a full 8×8 crosspoint switch. It is the fundamental building block of the CS–2 communications network. The route through the switch is determined by the header byte of each incoming message. Headers are added by the communications processor and removed by the switch as the message passes through it. In addition to a direct mapping from input link to output link, the switch supports broadcast and combining operations by mapping a single input to a contiguous range of outputs.