(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2007-0114690
(43) 공개일자 2007년12월04일

(51) Int. Cl.
G06F 9/46 (2006.01)
(21) 출원번호 10-2007-7005625
(22) 출원일자 2007년03월09일
심사청구일자 없음
번역문제출일자 2007년03월09일
(86) 국제출원번호 PCT/JP2006/307775
국제출원일자 2006년04월12일
(87) 국제공개번호 WO 2006/109835
국제공개일자 2006년10월19일
(30) 우선권주장
JP-P-2005-00114133 2005년04월12일 일본(JP)
JP-P-2005-00309352 2005년10월25일 일본(JP)

(71) 출원인
마츠시타 덴끼 산교 가부시키가이샤
일본 오오사카후 가도마시 오오아자 가도마 1006
(72) 발명자
모리시타 히로유키
일본국 오오사카후 가도마시 오오아자가도마 1006 마츠시타 덴끼산교 가부시키가이샤나이
하시모토 다카시
일본국 오오사카후 가도마시 오오아자가도마 1006 마츠시타 덴끼산교 가부시키가이샤나이
기요하라 도쿠조
일본국 오오사카후 가도마시 오오아자가도마 1006 마츠시타 덴끼산교 가부시키가이샤나이
(74) 대리인
김영철

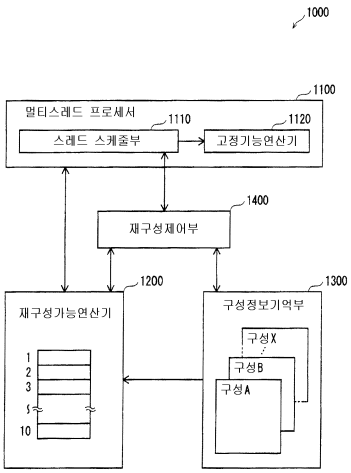
전체 청구항 수 : 총 5 항

(54) 프로세서

(57) 요 약

본 발명의 프로세서는, 복수의 스레드(thread)를 순회 적으로, 그 스레드에 할당된 시간씩 실행하는 프로세서로, 재구성 가능한 집적회로를 구비하며, 복수의 스레드 각각에 대응하는 회로구성정보를 기억해 두고, 회로구성정보에 의거하여 상기 집적회로의 일부분을 재구성하여, 스레드에 대응하는 회로구성정보에 의거하여 재구성된 집적회로를 이용하여 당해 스레드를 순차 실행시킨다. 어떤 스레드가 실행되고 있는 사이에 다음에 실행하는 스레드를 선택하여, 실행하고 있는 스레드가 사용하고 있는 상기 집적회로의 부분 이외의 부분에 대해서 다음에 실행하는 스레드를 위해서 재구성을 한다.

대표도 - 도1



특허청구의 범위

청구항 1

복수의 스레드(thread)를 순회 적으로(cyclically), 그 스레드에 할당된 시간씩 실행하는 프로세서로,

재구성 가능한 집적회로(reconfigurable integrated circuit)와,

회로구성정보(circuit configuration information set)에 의거하여 상기 집적회로의 일부분을 재구성하는 재구성수단과,

복수의 스레드 각각에 대응하는 회로구성정보를 기억하는 구성정보 기억수단과,

스레드에 대응하는 회로구성정보에 의거하여 재구성된 집적회로를 이용하여 당해 스레드를 순차 실행시키는 제어수단과,

상기 제어수단이 어떤 스레드를 실행시키고 있는 사이에 다음에 실행하는 스레드를 선택하는 선택수단을 구비하는 것을 특징으로 하는 프로세서.

청구항 2

청구항 1에 있어서,

상기 제어수단은, 스레드를 실행시키고 있는 사이에, 실행시키고 있는 스레드가 사용하고 있는 상기 집적회로의 부분 이외의 부분에 대해서, 상기 재구성수단으로 하여금 상기 선택수단에서 선택된 스레드에 대응하는 회로구성정보에 의거한 재구성을 더 하게 하는 것을 특징으로 하는 프로세서.

청구항 3

청구항 1에 있어서,

상기 프로세서는 연산기(computing unit)를 더 포함하며,

상기 제어수단은 상기 연산기와 상기 재구성된 집적회로를 이용하여 당해 스레드를 실행시키는 것을 특징으로 하는 프로세서.

청구항 4

복수의 명령으로 이루어지는 프로그램을 실행하는 프로세서로,

재구성 가능한 집적회로와,

회로구성정보에 의거하여 상기 집적회로의 일부분을 재구성하는 재구성수단과,

복수의 명령 각각에 대응하는 회로구성정보를 기억하는 구성정보 기억수단과,

상기 회로구성정보에 의거하여 상기 집적회로를 동시에 재구성할 수 있는 2 이상의 명령을 선택하는 선택수단과,

상기 선택수단에서 선택된 2 이상의 명령에 대응하는 회로구성정보에 의거하여 재구성된 집적회로를 이용하여 당해 2 이상의 명령을 병렬로 실행하는 실행수단을 구비하는 것을 특징으로 하는 프로세서.

청구항 5

재구성 가능한 집적회로를 가지며, 복수의 스레드를 순회 적으로, 그 스레드에 할당된 시간씩 실행하는 프로세서에서 사용되는 스레드 실행방법으로,

회로구성정보에 의거하여 상기 집적회로의 일부분을 재구성하는 재구성 스텝과,

복수의 스레드 각각에 대응하는 회로구성정보를 메모리에 기억하는 구성정보 기억스텝과,

스레드에 대응하는 회로구성정보에 의거하여 재구성된 집적회로를 이용하여 당해 스레드를 순차 실행시키는 제어스텝과,

상기 제어스텝이 어떤 스레드를 실행시키고 있는 사이에 다음에 실행하는 스레드를 선택하는 선택스텝을 구비하는 것을 특징으로 하는 스레드 실행방법.

명 세 서

기 술 분 야

<1> 본 발명은 프로세서에 관한 것으로, 특히, 재구성 가능한 집적회로를 갖는 프로세서에 관한 것이다.

배 경 기 술

<2> 최근의 프로세서, 예를 들어 디지털신호를 이용한 영상 및 음향기기에 탑재되는 프로세서는 복수의 처리에 대응할 필요가 있다.

<3> 영상을 압축하는 경우를 보면, 그 압축방법으로 MPEG(Moving Picture Experts Group) 2, MPEG 4, H.263, H.264 등, 다수의 규격이 실용화되어 있다.

<4> 따라서 사용자의 편리성 등을 고려하여, 최근의 영상 및 음향기기는 하나의 기기로 이들 복수의 규격에 대응하는 등의 복수의 기능의 실현이 요구된다.

<5> 이 요구에 응하기 위해서는, 하나의 처리를 하는 하드웨어를 복수 탑재함으로써 복수의 처리를 실현하거나, 또는, 하드웨어는 하나 탑재하고 소프트웨어에 의해 복수의 처리를 실행하는 방법을 생각할 수 있다.

<6> 전자의 방법은 고성능을 실현할 수 있다는 이점이 있으나, 실현해야 할 기능이 다수인 경우에는 회로규모가 커진다는 결점이 있다. 또, 새로 기능을 추가하는 경우 등에는 하드웨어의 추가가 필요해지게 된다.

<7> 한편, 후자의 방법은 소프트웨어의 추가나 변경 등에 의해 복수의 기능의 실현이나 추가 등을 유연하게 할 수 있다는 이점이 있으나, 성능을 향상시키기가 곤란하다는 결점이 있다.

<8> 그래서 균질의 회로 구성의 일부에 특정 처리에 적합한 회로를 포함시켜서, 동적으로 하드웨어 구성을 변경함으로써, 특정한 처리에 관해서 유연하면서도 고성능의 처리를 실현하는 재구성 가능한 하드웨어가 제안되어 있다(특허문헌 1 참조).

<9> 특허문헌 1 : 국제공개 제2002/095946호 팸플릿

<10> 그러나 이와 같은 재구성 가능한 하드웨어는 회로기능을 실장 하는 부분 외에도 배선부분 및 스위치 등도 필요하므로 회로규모가 커지지 않을 수 없고, 또, 재구성을 위해서는 시간을 필요로 한다.

<11> 그래서 본 발명은 회로규모를 억제하면서 유연하고도 고성능인 프로세서를 제공하는 것을 목적으로 한다.

발명의 상세한 설명

<12> 상기 과제를 해결하기 위해, 복수의 스레드(thread)를 순회 적으로cyclically), 그 스레드에 할당된 시간씩 실행하는 프로세서로, 재구성 가능한 집적회로(reconfigurable integrated circuit)와, 회로구성정보(circuit configuration information set)에 의거하여 상기 집적회로의 일부분을 재구성하는 재구성수단과, 복수의 스레드 각각에 대응하는 회로구성정보를 기억하는 구성정보 기억수단과, 스레드에 대응하는 회로구성정보에 의거하여 재구성된 집적회로를 이용하여 당해 스레드를 순차 실행시키는 제어수단과, 상기 제어수단이 어떤 스레드를 실행시키고 있는 사이에 다음에 실행하는 스레드를 선택하는 선택수단을 구비하는 것을 특징으로 한다.

<13> 본 발명의 프로세서는 상기 구성을 구비함으로써, 스레드별로 회로를 재구성할 수 있으므로, 그 스레드에 적합한 회로를 이용하여 실행할 수 있게 된다.

<14> 또, 상기 제어수단은, 스레드를 실행시키고 있는 사이에, 실행시키고 있는 스레드가 사용하고 있는 상기 집적회로의 부분 이외의 부분에 대해서, 상기 재구성수단으로 하여금 상기 선택수단에서 선택된 스레드에 대응하는 회로구성정보에 의거한 재구성을 더 하게 하는 것으로 해도 좋다.

<15> 이에 의해, 스레드를 실행하는 중에 다음 스레드를 위한 재구성을 할 수 있게 되므로, 재구성에 요하는 시간이 불필요해져서 유연하면서도 고성능인 프로세서를 실현할 수 있게 된다.

<16> 또, 상기 프로세서는 연산기(computing unit)를 더 포함하며, 상기 제어수단은 상기 연산기와 상기 재구성된 집

적회로를 이용하여 당해 스레드를 실행시키는 것으로 해도 좋다.

<17> 이에 의해, 통상의 연산기와 재구성한 집적회로를 사용하여 스레드를 실행할 수 있고, 처리에 따라서, 통상의 연산기를 사용하거나, 재구성한 연산기를 사용하거나, 또는 쌍방을 사용할 수 있으므로, 회로규모를 억제하면서, 유연하고도 고성능인 처리를 할 수 있게 된다. 예를 들어 재구성한 집적회로를 특정 처리용의 연산기로 사용하는 등이다.

<18> 즉, 처리를 위한 모든 집적회로를 재구성하는 것은 아니므로, 재구성 가능한 회로의 규모를 억제하여 프로세서 전체의 규모를 작게 할 수 있고, 또한, 처리에 필요한 회로는 재구성할 수 있으므로 유연하면서도 고성능의 처리를 할 수 있다.

<19> 또, 복수의 명령으로 이루어지는 프로그램을 실행하는 프로세서로, 재구성 가능한 집적회로와, 회로구성정보에 의거하여 상기 집적회로의 일부분을 재구성하는 재구성수단과, 복수의 명령 각각에 대응하는 회로구성정보를 기억하는 구성정보 기억수단과, 상기 회로구성정보에 의거하여 상기 집적회로를 동시에 재구성할 수 있는 2 이상의 명령을 선택하는 선택수단과, 상기 선택수단에서 선택된 2 이상의 명령에 대응하는 회로구성정보에 의거하여 재구성된 집적회로를 이용하여 당해 2 이상의 명령을 병렬로 실행하는 실행수단을 구비하는 것을 특징으로 한다.

<20> 본 발명의 프로세서는, 상기 구성을 구비함으로써 명령 별로 집적회로를 재구성할 수 있고, 재구성하는 회로의 규모 등에 따라서 복수 명령용의 재구성을 동시에 하므로, 회로규모를 억제하면서 유연하면서도 고성능의 처리가 가능해진다. 재구성을 동시에 하기 위해서는, 명령의 순서를 고려하는 것만이 아니라, 어떤 명령에 필요한 집적회로의 회로규모와 다른 명령에 필요한 집적회로의 회로규모를 합해도 재구성 가능한 논리회로의 회로규모를 초과하지 않을 필요가 있다.

실 시 예

<49> <실시 예 1>

<50> <개요>

<51> 본 발명의 프로세서는 통상의 프로세서가 구비하고 있는 연산부 이외에, 재구성 가능한 하드웨어(reconfigurable hardware)를 구비하여 처리를 분담함으로써, 회로규모를 억제하면서 고성능의 처리를 실현하는 것이다.

<52> 즉, 복수의 처리라고 하더라도 처리 전체가 모두 다르지는 않으며, 공통되는 처리나 명령, 빈번하게 사용되는 처리나 명령이 있다는 점에 착안하여, 공통되는 것은 통상의 연산장치에 의해서 행하고, 각 처리에 특유한 처리는 재구성 가능한 하드웨어에 의해 구성된 연산장치에 의해서 행하도록 함으로써, 프로세서 전체로서의 회로규모를 억제하면서 고성능을 유지하는 것이다.

<53> 본 실시 예의 프로세서는 멀티스레드 프로세서(multithread processor)이며, 멀티스레드를 실현하기 위한 방법으로 각 태스크(task)를 일정 시간씩 순번으로 실행해 가는 라운드 로빈 방식(round-robin method)을 취하는 것이다.

<54> 각 스레드(thread)는 각각에 고유한 회로가 있고, 고유한 회로를 필요로 하는 처리는 재구성 가능한 하드웨어로 재구성한 회로를 사용하여 실행한다.

<55> 즉, 본 프로세서는 스레드 각각에 고유한 회로를 구비할 필요가 없으므로, 전체 회로규모를 억제할 수 있다.

<56> 단, 재구성에는 시간을 요하므로 그 시간을 줄이는 것이 포인트가 된다.

<57> 이하, 본 발명의 실시 예의 멀티스레드 프로세서에 대하여 설명한다.

<58> <구성>

<59> 이하, 도 1을 이용하여 본 발명의 프로세서(1000)의 구성을 설명한다.

<60> 프로세서(1000)는 멀티스레드 프로세서(1100), 재구성가능 연산기(1200), 구성정보 기억부(1300) 및 재구성 제어부(1400)로 구성된다.

<61> 멀티스레드 프로세서(1100)는 통상의 프로세서이며, 복수의 다른 처리를 시분할(time sharing)로 실행할 수 있는 이른바 멀티스레드 프로세서이다.

<62> 멀티스레드 프로세서(1100)는 통상의 연산기(computing unit)인 고정기능 연산기(1120)와 스레드 스케줄부(thread scheduling unit, 1110)를 가지며, 스레드 스케줄부(1110)는 다음에 실행하는 스레드를 결정하는 등, 스레드의 스케줄링 기능을 갖는다. 이 스레드 스케줄부(1110)는 다음에 실행하는 스레드를 결정하여 레지스터의 저장·복귀(saving/restoring) 등의 준비를 행하는 외에, 본 발명에 특유한 처리도 한다.

<63> 구체적으로는, 스레드 실행 중에 다음에 실행하는 스레드를 선택하고, 그 스레드를 재구성 제어부(1400)에 통지하는 등이다.

<64> 이 멀티스레드 프로세서(1100)는 내부의 고정기능 연산기(1120)와 외부의 재구성가능 연산기(1200) 쌍방과 필요에 따라서 연산데이터의 송수신을 하면서 처리를 실행해 간다.

<65> 다음에, 재구성가능 연산기(1200)는 조합회로(combinational circuit) 및 순서 회로(sequential circuit)를 실현할 수 있는 논리블록(logical block)과, 논리블록 사이의 배선부분으로 이루어진다. 논리블록은 룩업 테이블(look-table)과 플립플롭을 포함하는 회로 유닛이며, 룩업 테이블의 설정치를 바꿈으로써 원하는 논리회로를 실현한다. 또, 배선부분에는 트랜지스터 스위치 등이 배치되며, 배선경로를 자유로 설정할 수 있게 되어 있다.

<66> 본 실시 예에서는 논리블록은 모두 동일한 구성으로 하고, 개개의 기능을 변경할 수 있으며, 이들을 각각 다르게 조합할 수 있는 배선 군으로 접속함으로써 각종 기능의 회로를 실현한다.

<67> 또, 재구성가능 연산기(1200)는 동일한 구성의 10개의 영역으로 분할되어 있는 것으로 한다. 이들 영역은 각각 독립해서 재구성을 할 수 있고, 영역 사이는 각각 다르게 조합할 수 있는 배선이 인출되며, 복수의 영역으로 하나의 회로를 실현할 수 있는 것으로 한다.

<68> 구성정보 기억부(1300)는 재구성가능 연산기(1200)를 원하는 회로로 재구성하기 위한 구성정보를 기억하는 기능을 갖는다. 구성정보는 원하는 회로의 개수만큼 있는 것으로 한다.

<69> 이 구성정보는 논리블록의 룩업 테이블의 설정치 및 배선경로를 설정하기 위한 각 트랜지스터 스위치에 대한 제어신호의 정보를 포함하는 것이다.

<70> 또, 구성정보 기억부(1300)는 구성정보 이외에도, 나중에 설명하는 스레드정보 테이블(1410)을 기억해 두는 기능을 갖는다. 이 테이블은 스레드와 그 스레드에서 사용하는 구성정보를 서로 대응시키고 있는 것이다.

<71> 재구성 제어부(1400)는 스레드 실행 중에 스레드 스케줄부(1110)로부터 다음 스레드의 통지를 받아서, 통상의 고정기능 연산기(1120) 만으로 실행할 수 있는가, 재구성가능 연산기(1200)를 필요로 하는가를 판단하여, 재구성가능 연산기(1200)를 재구성하는 경우에는 재구성가능 연산기(1200)와 구성정보 기억부(1300)에 지시를 하는 기능을 갖는다.

<72> 재구성가능 연산기(1200)에 대해서 재구성을 한다는 취지와 재구성을 하는 영역을 통지하고, 구성정보 기억부(1300)에는 구성정보를 지정하여, 그 구성정보를 재구성가능 연산기(1200)에 공급하도록 지시한다.

<73> 또, 재구성이 불가능한 경우에는 그 취지를 스레드 스케줄부(1110)에 알리는 기능을 갖는다. 재구성이 불가능한 경우란, 재구성가능 연산기(1200)에 재구성을 할 수 있는 영역이 없는 경우이다.

<74> <동작>

<75> 다음에, 도 2 내지 4를 이용하여 본 발명의 프로세서의 동작을 설명한다.

<76> 도 2를 이용하여 실행순서대로 스레드를 실행하는 경우의 예를 설명하고, 도 3을 이용하여 스레드의 실행순서를 바꾸는 경우의 예를 설명한다.

<77> 마지막으로, 도 4를 이용하여 스레드의 제어처리를 플로차트를 이용하여 설명한다.

<78> <실행순서대로 스레드를 실행하는 경우>

<79> 도 2 (a)는 스레드정보 테이블(1410)의 구성 예 및 내용 예를 나타내는 도면이고, 도 2 (b)는 스레드의 실행 예를 나타내는 타임차트이다.

<80> 도 2 (a)에 도시한 스레드정보 테이블(1410)의 스레드를 실행한 경우의 타임차트가 도 2 (b)에 도시한 타임차트이다.

<81> 먼저, 도 2 (a)의 스레드정보 테이블(1410)을 설명한다.

<82> 이 스레드정보 테이블(1410)은 구성정보 기억부(1300)에 기억되어 있는 것이다.

<83> 스레드정보 테이블(1410)은 스레드 명(1411), 구성정보(1412) 및 사용영역 수(1413)로 구성된다.

<84> 스레드 명(1411)은 스레드의 식별자이다. 이하, 「TH0」~「TH3」으로 표시되는 4개의 스레드가 순서대로 실행되는 것으로 하여 설명한다.

<85> 구성정보(1412)는 스레드 명(1411)에 의해 표시되는 스레드가 재구성가능 연산기(1200)를 사용하는 경우의 재구성을 위한 구성정보를 나타내고 있다.

<86> 사용영역 수(1413)는 재구성가능 연산기(1200)를 사용하는 경우에 필요한 영역 수를 나타낸다.

<87> 예를 들어, 스레드 명(1411) 「TH0」인 스레드는 구성정보(1412) 「구성 A」인 구성으로 재구성된 재구성가능 연산기(1200)를 사용하고, 구성정보(1412) 「구성 A」로 재구성가능 연산기(1200)를 재구성하는 데에는 사용영역 수(1413) 「6」 개의 영역이 필요하다. 또, 스레드 명(1411) 「TH1」에 대응하는 구성정보(1412) 「-」는 재구성가능 연산기(1200)를 사용하지 않는다는 취지를 나타내며, 당연히 사용영역 수(1413)는 「0」 개이다.

<88> 다음에, 도 2 (b)를 이용하여 스레드의 실행 예를 설명한다.

<89> 여기에서는, 스레드가 고정기능 연산기(1120)를 사용하고 있는 타임차트 10과, 그 스레드가 재구성가능 연산기(1200)를 사용하고 있는 경우의 구성정보를 나타내는 타임차트 20, 및 그 스레드가 실행 중에 재구성가능 연산기(1200)를 재구성하고 있는 구성정보를 나타내는 타임차트 30을 도시하고 있다. 또, 시간을 나타내는 화살표의 상방에는 구성정보를 나타내고, 하방에는 구성정보가 필요로 하는 영역 수를 나타낸다. 또한, 본 도 2 (b)에서는 모든 스레드는 고정기능 연산기(1120)를 사용하는 경우를 도시하고 있으나, 사용하지 않는 기간이 있어도 좋음은 물론이다.

<90> 먼저, 스레드 명 「TH0」, 「TH1」, 「TH2」, 「TH3」 순으로 스레드가 실행되며, 각각의 스레드의 실행 중에 다음의 스레드가 사용하는 재구성가능 연산기(1200)를 재구성해 두는 것으로 한다.

<91> 이와 같이 사전에 준비를 함으로써 재구성에 요하는 시간이 불필요해져서 실질적으로 스레드의 실행시간만으로 실행이 된다.

<92> 예를 들어, 스레드 명 「TH1」인 스레드 100이 고정기능 연산기(1120) 만을 사용하여 스레드를 실행하고 있다.

<93> 이 사이에, 다음에 실행 예정인 스레드 명(1411) 「TH2」인 스레드 110이 사용하는 구성정보(1412) 「구성 C」로 재구성가능 연산기(1200)를 재구성한다.

<94> 이 경우, 재구성가능 연산기(1200)는 모두 10개의 영역을 구비하고 있으므로, 사용하고 있는 재구성가능 연산기(1200)의 영역 수는 「구성 C」에 필요한 「3/10」 영역 101이 된다.

<95> 마찬가지로, 스레드 명 「TH2」인 스레드 110이 고정기능 연산기(1120)와 재구성가능 연산기의 「구성 C」를 사용하여 스레드를 실행하고 있다.

<96> 이 사이에, 다음에 실행 예정인 스레드 명(1411) 「TH3」인 스레드가 사용하는 구성정보(1412) 「구성 D」로 재구성가능 연산기(1200)를 재구성한다.

<97> 이 경우, 사용하고 있는 재구성가능 연산기(1200)의 영역 수는 「구성 C」가 사용하고 있는 「3/10」 영역과 「구성 D」가 필요로 하는 「4/10」 영역을 합한 「7/10」영역 111이 된다.

<98> 이와 같이, 순차로 스레드 실행 전에 필요한 재구성을 사전에 한다.

<99> <스레드의 실행순서를 바꾸는 경우>

<100> 다음에, 도 3을 이용하여, 스레드의 실행순서를 바꿈으로써 스레드의 실행 중에 다음 스레드를 위한 재구성이 가능해지는 예를 설명한다.

<101> 도 3 (a)는 스레드정보 테이블(1420)의 구성 예 및 내용 예를 나타내는 도면이고, 도 3 (b) 및 (c)는 스레드의 실행 예를 나타내는 타임차트이다.

<102> 도 3 (a)에 도시한 스레드정보 테이블(1420)의 스레드를 실행한 경우의 타임차트가 도 2 (b) 및 (c)에 도시한 타임차트이다.

<103> 먼저, 도 3 (a)의 스레드정보 테이블(1420)을 설명한다.

<104> 도 3 (a)의 스레드정보 테이블 1420은 도 2 (a)의 스레드정보 테이블 1410과 거의 동일하므로, 차이만을 설명한다.

<105> 다른 점은, 스레드 명(1411) 「TH1」의 스레드(1421)가 재구성가능 연산기(1200)를 사용하는 점이다. 구성정보(1412)가 「구성 B」, 사용영역 수(1413)가 「5」 개이다.

<106> 다음에, 도 3 (b)를 이용하여 스레드의 실행 예를 나타내는 타임차트를 설명한다. 타임차트가 의미하는 바는 도 2 (b)와 동일하다.

<107> 스레드 명 「TH0」인 스레드 200이 고정기능 연산기와 재구성가능 연산기의 「구성 A」를 사용하여 스레드를 실행하고 있다.

<108> 이 사이에, 다음에 실행 예정인 스레드 명(1411) 「TH1」인 스레드가 사용하는 구성정보(1412) 「구성 B」로 재구성가능 연산기(1200)를 재구성하려고 한다.

<109> 이 경우, 사용하고 있는 재구성가능 연산기(1200)의 영역 수는 「구성 A」가 사용하고 있는 「6/10」 영역과 「구성 B」가 필요로 하는 「5/10」 영역을 합한 「11/10」영역 201이 되어, 스레드 명 「TH0」인 스레드의 실행 중에 「구성 B」의 재구성을 할 수 없다.

<110> 그러므로 도 3 (c)에 도시하는 바와 같이, 스레드 명 「TH0」인 스레드 230 앞에 「TH2」인 스레드 220을 실행한다. 즉, 실행순서를 바꿔서 스레드를 실행한다.

<111> 그렇게 하면, 스레드 명 「TH0」인 스레드의 실행 중에 재구성하는 것은 스레드 명 「TH2」인 스레드 220이 사용하는 「구성 B」이며, 「구성 A」가 사용하고 있는 「6/10」 영역과 「구성 C」가 필요로 하는 「3/10」을 합한 「9/10」 영역 211이 되어, 사전에 재구성을 할 수 있게 된다.

<112> 마찬가지로, 스레드 명 「TH2」인 스레드 220이 고정기능 연산기(1120)와 재구성가능 연산기(1200)의 「구성 C」를 사용하여 스레드를 실행하고 있는 동안에, 다음에 실행 예정인 스레드 명(1411) 「TH1」인 스레드가 사용하는 구성정보(1412) 「구성 B」로 재구성가능 연산기(1200)를 재구성한다.

<113> 이 경우, 사용하고 있는 재구성가능 연산기(1200)의 영역 수는 「구성 C」가 사용하고 있는 「3/10」 영역과 「구성 B」가 필요로 하는 「5/10」 영역을 합한 「8/10」영역 222가 된다.

<114> 통상, 라운드 로빈 방식에서는 각 스레드에서 실행해야할 처리에 따라서 스레드에 할당되는 타임 슬라이스(time slice)의 길이가 고려된다. 즉, 처리시간을 보장하여야 하는 경우 등에는 타임 슬라이스가 긴 스레드를 할당하는 등이다.

<115> 따라서 주기가 깨어지지 않을 것이 처리를 스레드에 할당할 때의 전제조건이 된다.

<116> 그러나 하나의 타임 슬라이스의 시간은 처리 전체의 실행시간에 비하면 매우 작은 것이므로, 본 프로세서에서는 소정 시간 내에서 각 스레드의 실행시간이 예정대로 이루어지도록 스레드 스케줄부(1110)에서 조정을 하는 것으로 한다. 예를 들어, 모든 10회 실행하는 것을 1 단위로 하여 각 스레드의 실행 횟수를 카운트하여 두고, 최초의 스레드가 11회째의 실행을 개시하기 전에 다른 스레드에서 10회에 이른 것은 우선적으로 실행한다. 모든 스레드가 10회 실행한 후에 최초의 스레드의 11회째 실행을 개시하는 등이다.

<117> <스레드의 제어처리>

<118> 다음에, 도 4를 이용하여 본 프로세서의 스레드 제어의 처리에 대해서 설명한다.

<119> 도 4는 본 발명의 스레드 제어의 처리를 나타내는 플로차트이다.

<120> 스레드 스케줄부(1110)가 다음에 실행하는 스레드를 선택한다(스텝 S 100). 제어처리를 개시한 직후인 경우에는 최초의 스레드이다.

<121> 각 스레드에 할당되어 있는 처리가 모두 종료되어 있는 경우에는(스텝 S 110 : Y) 스레드의 제어처리를 종료한다.

<122> 선택된 스레드인 다음 스레드를 실행하는 경우에는 그 스레드 명(1411)을 재구성 제어부(1400)에 인계하여 재구성을 의뢰한다.

<123> 의뢰를 받은 재구성 제어부(1400)는 수신한 스레드 명(1411)이 재구성가능 연산기(1200)를 사용하는가 여부를 구성정보 기억부(1300)에 기억되어 있는 스레드정보 테이블(1410)을 참조하여 판단한다. 구체적으로는, 수신한

스레드 명(1411)에 대응하는 구성정보(1412)에 구성이 지정되어 있는 경우에는 재구성가능 연산기(1200)를 사용하는 것으로 판단한다.

<124> 사용하지 않는다고 판단한 경우에는(스텝 S 120 : N), 재구성 제어부(1400)는 그 취지를 스레드 스케줄부(1110)에 알리고, 스레드 스케줄부(1110)는 현재 실행중인 스레드가 종료하는 대로 다음 스레드의 실행을 개시한다(스텝 S 150).

<125> 한편, 사용하는 것으로 판단한 경우에는(스텝 S 120 : Y) 재구성하는 영역이 비어 있는가 여부를 판단한다. 구체적으로는, 수신한 스레드 명(1411)에 대응하는 사용영역 수(1413)에 제시되어 있는 수만큼의 영역이 비어 있는가를 판단한다.

<126> 재구성 제어부(1400)는 내부에 현재 사용되고 있는 영역의 번호를 기억하고 있는 것으로 하고, 사용하고 있는 스레드의 타임 슬라이스가 종료한 경우에는 사용하고 있던 영역이 빈 것이 되며, 기억하고 있는 영역번호에서 소거한다.

<127> 영역이 비어 있지 않은 것으로 판단한 경우에는(스텝 S 130 : N) 재구성 제어부(1400)는 스레드 스케줄부(1110)에 그 취지를 통지한다. 스레드 스케줄부(1110)는 다른 스레드를 선택한다(스텝 S 100). 스레드 스케줄부(1110)는 각 스레드의 실행 횟수를 기억해 두고, 적절한 때에 우선적으로 선택하여 전체 스레드의 실행 횟수를 합치는 것으로 한다.

<128> 한편, 영역이 비어 있다고 판단한 경우에는(스텝 S 130 : Y), 재구성 제어부(1400)는 재구성가능 연산기(1200)에 재구성을 한다는 취지를 통지하고, 구성정보 기억부(1300)에는 수신한 스레드 명(1411)에 대응하는 구성정보(1412)를 영역을 지정해서 송신하도록 한다. 송신 후, 재구성 제어부(1400)는 내부에 기억하고 있는 사용 중인 영역번호를 갱신한다.

<129> 재구성가능 연산기(1200)는 구성정보 기억부(1300)에서 송신된 구성정보로 재구성을 행하고(스텝 S 140), 완료하면 재구성 제어부(1400)에 통지한다.

<130> 재구성이 완료하였다는 취지의 통지를 받은 재구성 제어부(1400)는 그 취지를 스레드 스케줄부(1110)에 알리고, 스레드 스케줄부(1110)는 현재 실행 중인 스레드의 종료 후에 바로 스레드의 실행을 개시한다(스텝 S 150).

<131> 스레드를 개시하도록 한 스레드 스케줄부(1110)는 다음 스레드를 선택한다(스텝 S 100).

<132> <실시 예 2>

<133> <개요

<134> 실시 예 1이 스레드 별로 재구성한 재구성가능 연산기를 사용하는 것인데 반해, 본 실시 예에서는 명령 별로 재구성한 재구성가능 연산기를 사용하는 것이다.

<135> 이하, 본 실시 예 2의 구성 등을 설명한다.

<136> <구성>

<137> 도 5는 실시 예 2의 프로세서(500)의 구성 예를 나타내는 도면이다.

<138> 프로세서(5000)는 명령 페치부(5100), 명령 디코드부(5200), 연산 제어부(5300), 어드레스 테이블 기억부(5400), 재구성정보 기억부(5500), 재구성가능 연산기(5600) 및 고정기능 연산기(5700)를 구비하며, 외부에 명령 기억부(5010)를 구비한다.

<139> 명령 기억부(5010)는 프로세서(5000)에서 실행하는 명령 코드를 기억해 두는 기능을 갖는다.

<140> 명령 페치부(5100)는 명령 기억부(5010)에서 명령 코드를 판독하여 명령 디코드부(5200)에 인계하는 기능을 갖는다.

<141> 명령 디코드부(5200)는 명령 페치부(5100)로부터 명령 코드를 수신하고 해석하는 통상의 기능 이외에, 본 발명에 독자적인 기능을 갖는다.

<142> 구체적으로는, 디코드 결과, 재구성가능 연산기(5600)를 이용하는 명령인 경우에는 명령 종별에 구성정보가 기억되어 있는 어드레스를 어드레스 테이블 기억부(5400)에서 취득하고, 재구성정보 기억부(5500)에 어드레스를 인계하여 재구성가능 연산기(5600)에 구성정보를 송신하도록 하는 기능이다.

<143> 어드레스 테이블 기억부(5400)는 명령 종별과 그 구성정보의 어드레스를 서로 대응시켜서 기억하고 있는 기능을 갖는다.

<144> 연산 제어부(5300)는 명령 디코드부(5200)가 디코드 한 결과에 따라서 연산동작을 제어하는 기능을 갖는다. 고정기능 연산기(5700)와 재구성가능 연산기(5600)에 대해서 타이밍을 맞춰서 명령을 발행해 간다.

<145> 재구성정보 기억부(5500)는 복수의 각 명령에 대응하여 구성정보를 기억하는 기능을 갖는다. 여기서, 기억되어 있는 각 구성정보의 선두 어드레스가 어드레스 테이블 기억부(5400)에서 명령 종별과 대응시켜서 기억되어 있다. 이 구성정보는 실시 예 1의 구성정보 기억부(1300)에 기억되어 있는 구성정보와 동일하다.

<146> 또, 재구성정보 기억부(5500)는 명령 디코드부(5200)로부터의 지시에 의해서, 지정된 어드레스의 구성정보를 재구성가능 연산기(5600)에 송신하는 기능도 구비하고 있다.

<147> 재구성가능 연산기(5600)는 재구성을 할 수 있는 연산기이며, 실시 예 1의 재구성가능 연산기(1200)와 동일하다. 다만, 본 실시 예에서는 4개의 영역으로 분할되어 있는 것으로 한다.

<148> 고정기능 연산기(5700)는 복수의 고정기능 연산기로 구성되며, 본 실시 예에서는 3개의 고정기능 연산기(5701, 5702, 5703)로 구성되는 것으로 한다.

<149> 이하, 명령과 구성정보의 대응관계를 간단히 설명하고 동작을 설명한다.

<150> <명령코드와 구성정보의 대응관계>

<151> 본 발명에서 사용하는 명령코드로부터 그 명령의 실행에 필요한 재구성을 행하기 위한 구성정보를 구하는 방법을 도 6 및 도 7을 이용하여 설명한다.

<152> 도 6은 본 발명에서 사용하는 명령코드의 구성 예를 나타내는 도면이고, 도 7은 명령정보 테이블(5410)의 구성 예 및 내용 예를 나타내는 도면이다.

<153> 먼저, 도 6의 명령코드의 구성 예부터 설명한다.

<154> 본 발명에서 사용하는 명령코드(5110)는 명령의 종별을 나타내는 오피 코드(operand code, 5111)와 이 명령에서 취급하는 값 등을 나타내는 오퍼랜드(operand, 5112)로 구성된다.

<155> 본 발명에서는 이 오피 코드(5111)와 구성정보가 대응되어 있고(화살표 참조), 명령의 실행에 재구성가능 연산기(5600)가 필요한 것으로 프로세서에 의해서 판단된 경우에는, 대응되어 있는 구성정보에 의해서 재구성된 재구성가능 연산기(5600)에서 명령이 실행된다.

<156> 프로세서에 의해서 명령의 실행에 재구성가능 연산기(5600)이 불필요하다고 판단된 경우에는, 고정기능 연산기(5700)를 이용하여 실행이 이루어진다.

<157> 다음에, 도 7의 명령정보 테이블(5410)에 대해서 설명한다.

<158> 이 명령정보 테이블(5410)은 어드레스 테이블 기억부(5400)에 기억되어 있는 것으로 한다.

<159> 명령정보 테이블(5410)은 오피 코드 종별(5411), 어드레스(5412) 및 사용영역 수(5413)로 구성된다.

<160> 오피 코드 종별(5411)은 명령코드의 오피 코드, 즉, 명령을 나타내는 것이다. 여기에서는 재구성가능 연산기(5600)를 이용하는 명령만이 기재되어 있는 것으로 한다.

<161> 따라서 여기에 기재되어 있지 않은 명령은 고정기능 연산기(5700)에서 실행하는 것이 된다.

<162> 다음에, 어드레스(5412)는 오피 코드 종별(5411)에서 표시되는 오피 코드에 대응되어 있는 구성정보의 재구성정보 기억부(5500) 내에서의 어드레스를 나타낸다. 또한, 본 실시 예에서는 어드레스로 하고 있으나, ID 등, 재구성정보를 특정할 수 있는 것이면 된다.

<163> 사용영역 수(5413)는 재구성가능 연산기(5600)을 사용하는 경우의 영역 수를 나타낸다. 예를 들어, 오피 코드 종별(5411)이 「sub」인 명령은 어드레스(5412) 「addr 1」로 표시되는 어드레스에 기억되어 있는 구성정보로 재구성된 재구성가능 연산기(5600)를 사용하며, 재구성가능 연산기(5600)를 재구성하기 위해서는 사용영역 수(5413)가 「3」개인 영역이 필요하다.

<164> 본 실시 예의 경우, 명령을 사용하는 재구성가능 연산기(5600)의 영역 수를 고려하여, 프로그램을 기계어로 변환하는 컴파일 시에, 본 실시 예에서는 명령코드(5110)로 변환할 때에, 명령의 순서 및 명령에 의해 재구성하는

영역의 번호를 결정해 두는 것으로 한다. 즉, 컴파일 시에, 실행하는 명령의 순서를 고려한 상태에서, 앞의 명령의 실행 중에 재구성할 수 있는 명령 순으로 구성하여, 각각의 재구성을 하는 영역을 결정해 두는 것으로 한다. 또, 사용하는 영역번호는, 예를 들어 오퍼랜드에 의해 영역번호를 지정하거나, 명령 별로 영역번호를 정해 두는 등을 하여, 명령 디코드부가 알 수 있게 해 두는 것으로 한다.

<165> <동작>

<166> 다음에, 도 8 내지 10을 이용하여 어떻게 명령이 실행되는가를 설명한다.

<167> 도 8은 본 발명의 명령세트를 이용한 프로그램 예이고, 도 9는 프로그램을 동작시킨 프로세서의 파이프라인 동작(pipeline processing)의 예이다.

<168> 또, 도 10은 본 실시 예의 프로세서의 명령실행처리를 나타내는 플로차트이다.

<169> 먼저, 도 8의 프로그램에 대해서 간단히 설명한다.

<170> 오피 코드(5111) 「Add」, 오퍼랜드(5112) 「r0, r1, r2」인 명령코드는 레지스터 1의 내용과 레지스터 2의 내용을 가산하여 결과를 레지스터 0에 대입하는 것을 의미하고, 오피 코드(5111) 「Sub」, 오퍼랜드(5112) 「r3, r1, r3」인 명령코드는 레지스터 1의 내용에서 레지스터 3의 내용을 감산하여 결과를 레지스터 3에 대입하는 것을 의미한다.

<171> 또, 오피 코드(5111) 「Reconf 0」, 오퍼랜드(5112) 「r2, r0, 0xfe」인 명령코드는 레지스터 0의 내용과 즉치(immediate data) 「0xfe」를 이용하여 연산 「Reconf 1」을 하여 결과를 레지스터 2에 대입하는 것을 의미하며, 오피 코드(5111) 「Reconf 1」, 오퍼랜드(5112) 「r3, r1, r3」인 명령코드는 레지스터 1의 내용과 레지스터 3의 내용을 이용하여 연산 「reconf 1」을 하여 결과를 레지스터 3에 대입하는 것을 의미한다.

<172> 다음에, 도 9 및 도 10을 이용하여 프로그램을 실행하는 프로세서의 동작을 설명한다. 도 10의 플로차트에 따라서, 도 9의 타임차트를 참조하면서 설명한다.

<173> 도 8에 도시하는 프로그램이 명령 기억부(5010)에 기억되어 있는 것으로 한다.

<174> 먼저, 명령 페치부(5100)는 명령코드 「Add r0, r1, r2」를 페치(fetch) 하여(도 10 : 스텝 S 800, 도 9 : 스텝 S 500) 명령 디코드부(5200)에 인계한다.

<175> 명령코드를 수신한 명령 디코드부(5200)는 수신한 명령코드를 해석한다.

<176> 수신한 명령코드가 종료를 나타내는 취지의 코드인 경우에는(도 10 : 스텝 S 810 : Y) 명령 디코드부(5200)는 처리를 종료한다.

<177> 또, 명령코드가 종료를 나타내는 취지가 아닌 경우에는(도 10 : 스텝 S 810 : N) 명령 디코드부(5200)는 오피 코드(5111) 「Add」를 어드레스 테이블 기억부(5400)에 인계하여 구성정보의 어드레스를 요구한다.

<178> 어드레스 테이블 기억부(5400)는 명령정보 테이블(5410)을 참조하여 인계된 오피 코드(5111) 「Add」가 오피 코드 종별(5411)에 존재하는가를 검색하고, 그 결과, 오피 코드 종별(5411)에 없으면 재구성가능 연산기(5600)는 사용하지 않는다는 취지를 명령 디코드부(5200)에 알린다(도 10 : 스텝 S 820 : N, 도 9 : 스텝 S 510).

<179> 재구성가능 연산기(5600)를 사용하지 않는다는 취지를 수신한 명령 디코드부(5200)는 「Add r0, r1, r2」의 디코드 완료 명령을 연산 제어부(5300)에 인계한다.

<180> 디코드 결과를 인계받은 연산 제어부(5300)는 고정기능 연산기(5700)에 명령을 발행하여 명령 코드 「Add r0, r1, r2」를 실행한다(도 10 : 스텝 S 830, 840).

<181> 명령 페치부(5100)는 명령코드 「Add r0, r1, r2」를 페치한 후, 다음 명령코드 「Sub r3, r1, r3」을 페치하여(도 10 : 스텝 S 800, 도 9 : 스텝 S 520) 명령 디코드부(5200)에 인계한다.

<182> 명령코드를 수신한 명령 디코드부(5200)는 오피 코드(5111) 「Sub」를 어드레스 테이블 기억부(5400)에 인계하여 구성정보의 어드레스를 요구한다.

<183> 어드레스 테이블 기억부(5400)는 명령정보 테이블(5410)을 참조하여 오피 코드 종별(5411)에 인계된 오피 코드(5111) 「Sub」가 존재하는가를 검색하고, 그 결과, 존재하면 어드레스(5412) 「Addr 1」을 명령 디코드부(5200)에 송신한다(도 10 : 스텝 S 820 : Y, 도 9 : 스텝 S 600).

<184> 어드레스를 수신한 명령 디코드부(5200)는 재구성정보 기억부(5500)에 수신한 어드레스(5412) 「Addr 1」을 인계하고, 그 어드레스의 구성정보를 재구성가능 연산기(5600)에 송신하여 재구성을 행한다는 취지의 지시를 한다.

<185> 지시를 한 명령 디코드부(5200)는 「Sub r3, r1, r3」의 디코드 완료명령을 연산 제어부(5300)에 인계한다.

<186> 한편, 지시를 수신한 재구성정보 기억부(5500)는 수신한 어드레스의 구성정보를 재구성가능 연산기(5600)에 송신하여 재구성을 행한다(도 10 : 스텝 S 850, 도 9 : 스텝 S 610).

<187> 이 재구성은 재구성가능 연산기(5600)의 4 영역 중 3 영역을 재구성한다(도 7 참조).

<188> 디코드 결과를 인계받은 연산 제어부(5300)는 재구성가능 연산기(5600)에 명령을 발행하여 명령코드 「Sub r3, r1, r3」을 실행한다(도 10 : 스텝 S 860, 도 9 : 스텝 S 620).

<189> 그 후, 실행결과를 레지스터 3에 기록한다(도 9 : 스텝 S 630).

<190> 이와 같이, 명령을 순차 동시에 실행해 간다.

<191> 여기서, 명령코드 「Sub r3, r1, r3」의 다음의 명령코드 「Reconf 0 r2, r0, 0xfe」를 실행하는 경우를 보면, 이 오피 코드 「Reconf 0」은 재구성가능 연산기(5600)를 사용한다.

<192> 따라서 「Sub r3, r1, r3」을 실행하는 중(도 9 : 스텝 S 620)에 재구성을 행하는 것이다(도 9 : 스텝 S 700).

<193> 이 오피 코드 종별(5411) 「Reconf 0」은 사용영역 수(5413)가 「1」개이므로, 「Sub r3, r1, r3」이 영역을 3개 사용하여 실행중이라도 재구성이 가능하다.

<194> 다음의 명령코드 「Reconf 1 r3, r1, r3」을 실행하는 경우에도 동일하다.

<195> 이와 같이, 재구성 가능한 연산기를 갖는 프로세서에서 명령단위로 재구성 가능한 연산기를 제어할 수 있게 되어서, 유연하고도 고성능의 연산처리를 높은 면적효율(area-efficiency)로 실현할 수 있다.

<196> <변형 예>

<197> 다음에, 실시 예 2의 변형 예에 대해서 설명한다.

<198> 실시 예 2에서는 1 명령마다 재구성가능 연산기를 재구성을 하였으나, 본 변형 예에서는 복수의 명령을 한꺼번에 재구성하는 예를 설명한다. 동시에 복수의 명령을 실행할 수 있으므로 처리속도의 향상을 도모할 수 있다.

<199> 도 11 내지 13을 이용하여 어떻게 명령이 실행되는가를 설명한다.

<200> 도 11은 본 변형 예의 명령세트를 이용한 프로그램 예이고, 도 12는 프로그램을 동작시킨 프로세서의 파이프라인 동작의 예이다.

<201> 또, 도 13은 본 변형 예의 프로세서의 명령실행처리를 나타내는 플로차트이다.

<202> 먼저, 도 11의 프로그램의 각 명령의 내용에 대해서는 도 8과 동일하다.

<203> 다만, 「Reconf 0 r2, r0, 0xfe」와 「Reconf 1 r3, r1, r3」을 동일한 스테이지에서 행하는 점이 다르다. 도 11에서는 설명의 편의상 「Reconf 0 r2, r0, 0xfe」와 「Reconf 1 r3, r1, r3」을 횡으로 배열하여 기재함으로써 동시에 실행된다는 것을 표시하고 있으나, 2개의 명령을 동시에 실행한다는 것을 명령 디코드부(5200)가 해석할 수 있도록 컴파일러가 출력해 두는 것으로 한다.

<204> 예를 들어, 명령이 사용하는 재구성가능 연산기(5600)의 영역 수를 고려하여, 컴파일 시에 명령의 순서 및 명령에 의해 재구성하는 영역의 번호를 정한다. 즉, 컴파일 시에 명령실행의 순서를 고려하여, 동시에 실행 가능하면서 동시에 재구성을 할 수 있는 명령을 선택하여, 각각의 재구성을 행하는 영역을 정한다. 이 병렬로 실행하는 명령과 사용하는 영역번호는, 예를 들어 병렬실행을 나타내는 명령코드를 마련하여, 그 오퍼랜드에 이들 명령과 영역번호 등을 기재한다.

<205> 구체적으로는, 컴파일 시에 사용영역 수(5413)가 고려되며, 「Sub」와 「Reconf 0」은 동시에 재구성을 할 수 있으나, 「Sub」와 「Reconf 1」은 동시에 재구성을 할 수 없는 것이 된다. 「Sub」와 「Reconf 0」은 합계 4개의 영역을 사용하나, 「Sub」와 「Reconf 1」은 합계 5개의 영역을 필요로 하기 때문이다(도 7 참조).

<206> 도 12 및 도 13을 이용하여 프로그램을 실행하는 프로세서의 동작을 간단하게 설명한다. 도 13의 플로차트에 따

라서, 도 12의 타임차트를 참조하면서 설명한다.

<207> 도 13의 플로차트는 도 10의 플로차트의 차이점만을 설명한다. 구체적으로는, 스텝 S 900 내지 스텝 S 920까지이다.

<208> 「Reconf 0 r2, r0, 0xfe」와 「Reconf 1 r3, r1, r3」은 재구성가능 연산기를 사용하면서 동시에 실행한다고 해석한 명령 디코드부(5200)는 오피 코드(5111) 「Reconf 0」과 「Reconf 1」을 어드레스 테이블 기억부(5400)에 인계하여 구성정보의 어드레스를 요구한다.

<209> 어드레스 테이블 기억부(5400)는 명령정보 테이블(5410)을 참조하여 어드레스(5412) 「addr 4」와 「addr 5」를 명령 디코드부(5200)에 송신한다(도 13 : 스텝 S 820, : Y, 스텝 S 900 : Y).

<210> 어드레스를 수신한 명령 디코드부(5200)는 재구성정보 기억부(5500)에 수신한 어드레스(5412) 「addr 4」와 「addr 5」를 인계하고, 그 어드레스의 구성정보를 재구성가능 연산기(5600)에 송출하여 재구성을 행한다는 취지의 지시를 한다.

<211> 지시를 한 명령 디코드부(5200)는 「Reconf 0 r2, r0, 0xfe」와 「Reconf 1 r3, r1, r3」의 디코드 완료 명령을 연산 제어부(5300)에 인계한다.

<212> 한편, 지시를 수신한 재구성정보 기억부(5500)는 수신한 어드레스의 구성정보를 재구성가능 연산기(5600)에 송신하여 재구성을 행한다(도 13 : 스텝 S 910, 도 12 : 스텝 S 700).

<213> 디코드 결과를 인계받은 연산 제어부(5300)는 재구성가능 연산기(5600)에 명령을 발행하여 명령코드 「Reconf 0 r2, r0, 0xfe」와 「Reconf 1 r3, r1, r3」을 실행한다(도 13 : 스텝 S 920, 도 12 : 스텝 S 710).

<214> 명령코드를 수신한 명령 디코드부(5200)는, 재구성가능 연산기를 사용하지 않는 경우(도 13 : 스텝 S 820 : N)와 사용하였다고 해도 하나의 명령인 경우(도 13 : 스텝 S 900 : N)에는, 각각 도 10과 동일한 처리를 한다(도 13 : 스텝 S 840~스텝 S 870).

<215> 또한, 여기에서는 재구성 가능한 연산기를 사용하는 2개의 명령을 동시에 처리하는 경우를 설명하였으나, 동시에 발행할 수 있는 명령의 수는 2개로 한정되는 것은 아니다. 또, 고정기능 연산기(5700)를 사용하는 명령과 동시에 처리해도 좋다.

<216> <보충>

<217> 이상, 본 발명의 프로세서에 대해서 실시 예에 의거하여 설명하였으나, 이 프로세서를 부분적으로 변형할 수도 있으며, 본 발명은 앞에서 설명한 실시 예로 한정되는 것은 물론 아니다. 즉,

<218> (1) 실시 예 2에서는 명령코드의 오피 코드와 구성정보를 서로 대응시키고 있었으나, 이에 한정되지는 않는다.

<219> 예를 들어, 오퍼랜드의 일부에 구성정보를 나타내는 코드를 포함시켜 두어도 좋다. 도 14에 도시하는 바와 같이, 명령코드(5150)의 오퍼랜드의 필드(5150)에 구성정보의 ID 등을 포함시켜 두고, 실행시에 ID에 따라서 구성정보를 특정한다.

<220> (2) 실시 예 2에서는 설명의 편의상 복수의 고정기능 연산기의 실행에 대해서는 상세한 설명을 하고 있지 않으나, 이들 고정기능 연산기와 재구성가능 연산기가 동시에 실행할 수 있는 것인 경우에는 동시에 복수의 명령을 발행하는 것으로 해도 좋다.

<221> 동시에 발행하는 명령의 결정 여하에 따라서는 연산효율을 대폭적으로 향상시킬 수도 있게 된다.

<222> 즉, 재구성 가능한 하드웨어에 의해 구성된 연산기에서는 복수 종류의 연산기능을 선택적으로 실행할 수 있다. 따라서 본 발명에 의한 명령세트를 이용함으로써 명령의 병렬도를 향상시킨 최적의 기능을 실현하는 프로그램을 작성할 수 있게 되기 때문이다.

<223> 동시에 발행하는 명령을 결정하는 동작은, 예를 들어 프로세서의 내부에서 명령의 해석시에 이루어지는 것이라도, 프로세서에 대해서 이루어지는 프로그램 시점에서 미리 이루어지는 것이라도 좋다.

<224> (3) 실시 예에서는 재구성가능 연산기는 균질의 복수의 영역으로 구분되어 있는 것으로 하고 있으나, 영역별로 다른 논리블록으로 해도 좋고, 또 그 영역의 크기가 달라도 좋다.

<225> (4) 실시 예에서는 재구성가능 연산기를 구성하는 논리블록은 룩업 테이블과 플립플롭을 포함하는 회로유닛인 것으로 하고 있으나, 논리블록은 ALU(Arithmetic and Logical Unit), 시프트(shift), 데이터 제어 및 논리연산

을 하는 유닛, 플립플롭 등으로 구성된 것, 즉, 일반 논리회로의 조합으로 구성된 것이라도 좋다.

산업상 이용 가능성

<226> 본 발명의 프로세서는 회로규모를 억제하면서 유연하면서도 고성능의 처리를 실현할 수 있으므로, 화상처리 LSI의 연산기 등에 특히 유용하다.

도면의 간단한 설명

<21> 도 1은 프로세서(1000)의 구성 예를 나타내는 도면이다.

<22> 도 2 (a)는 스레드정보 테이블(1410)의 구성 예 및 내용 예를 나타내는 도면이고,

<23> 도 2 (b)는 스레드의 실행 예를 나타내는 타임차트이다.

<24> 도 3 (a)는 스레드정보 테이블(1420)의 구성 예 및 내용 예를 나타내는 도면이고,

<25> 도 2 (b) 및 (c)는 스레드의 실행 예를 나타내는 타임차트이다.

<26> 도 4는 본 프로세서의 스레드 제어의 처리를 나타내는 플로차트이다.

<27> 도 5는 실시 예 2의 프로세서(5000)의 구성 예를 나타내는 도면이다.

<28> 도 6은 실시 예 2에서 사용하는 명령코드의 구성 예를 나타내는 도면이다.

<29> 도 7은 명령정보 테이블(5410)의 구성 예 및 내용 예를 나타내는 도면이다.

<30> 도 8은 실시 예 2에 관한 명령세트를 이용한 프로그램 예이다.

<31> 도 9는 프로그램을 동작시킨 프로세서의 파이프라인 동작의 예이다.

<32> 도 10은 실시 예 2의 프로세서의 명령실행처리를 나타내는 플로차트이다.

<33> 도 11은 변형 예의 명령세트를 이용한 프로그램 예이다.

<34> 도 12는 변형 예의 프로그램을 동작시킨 프로세서의 파이프라인 동작의 예이다.

<35> 도 13은 변형 예의 프로세서의 명령실행처리를 나타내는 플로차트이다.

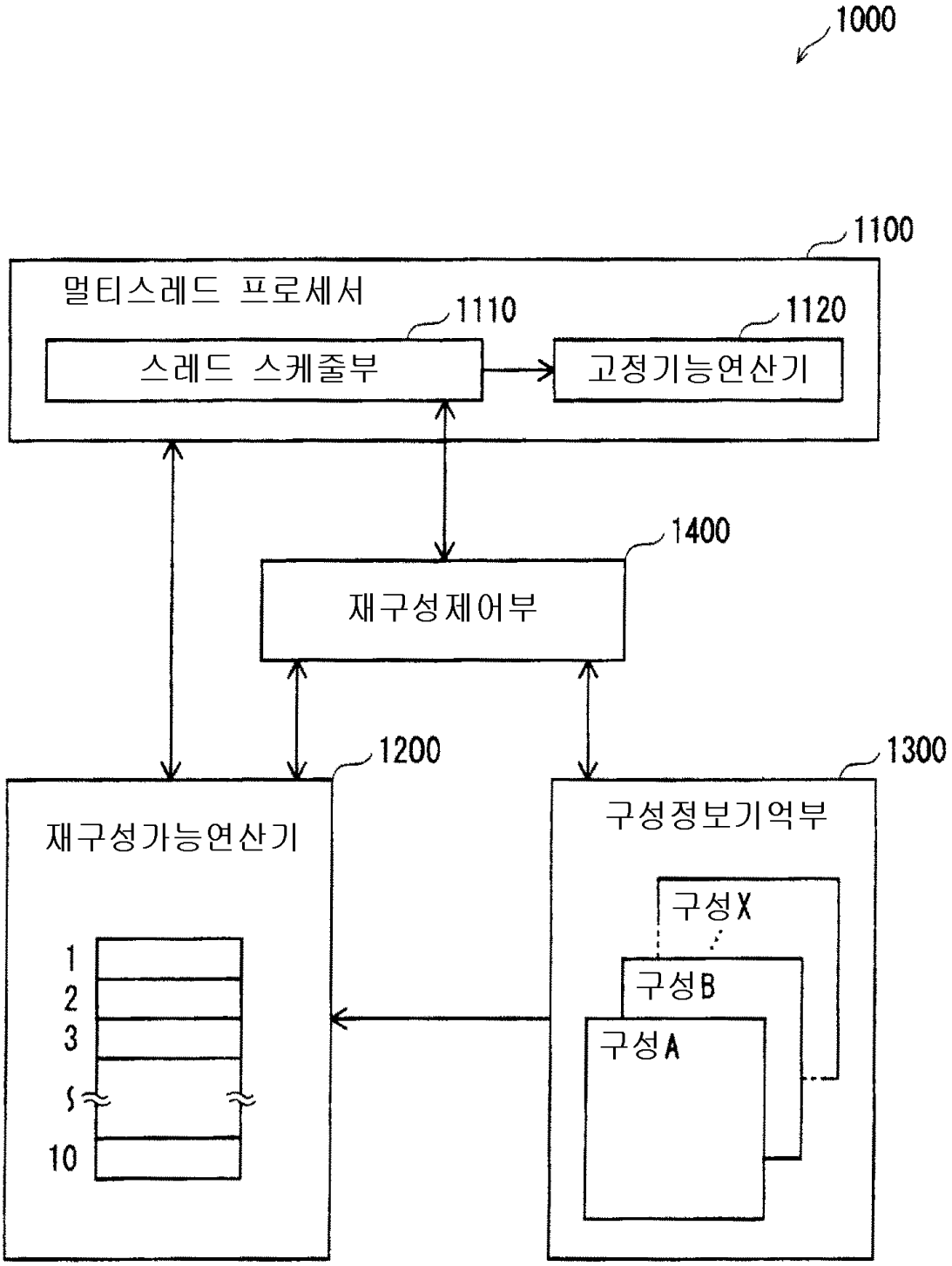
<36> 도 14는 실시 예 2에서 사용하는 명령코드의 구성의 변형 예를 나타내는 도면이다.

<37> (부호의 설명)

<38>	1000, 5000	프로세서	1100	멀티스레드 프로세서	
<39>	1110	스레드 스케줄부	1120	고정기능 연산기	
<40>	1200	재구성 가능 연산기	1300	구성정보 기억부	
<41>	1400	재구성 제어부	1410, 1420	스레드정보 테이블	
<42>	5010	명령 기억부	5100	명령 페치부	
<43>	5110, 5150	명령코드	5111	오피 코드	
<44>	5112	오퍼랜드	5200	명령 디코드부	
<45>	5300	연산제어부	5400	어드레스테이블 기억부	
<46>	5410	명령정보 테이블	5413	사용영역 수	
<47>	5500	재구성정보 기억부	5600	재구성가능 연산기	
<48>	5700	고정기능 연산기			

도면

도면1


1000
1100
멀티스레드 프로세서
1110
스레드 스케줄부
1120
고정기능연산기
1400
재구성제어부
1200
재구성가능연산기
1
2
3
S
10
1300
구성정보기억부
구성X
구성B
구성A

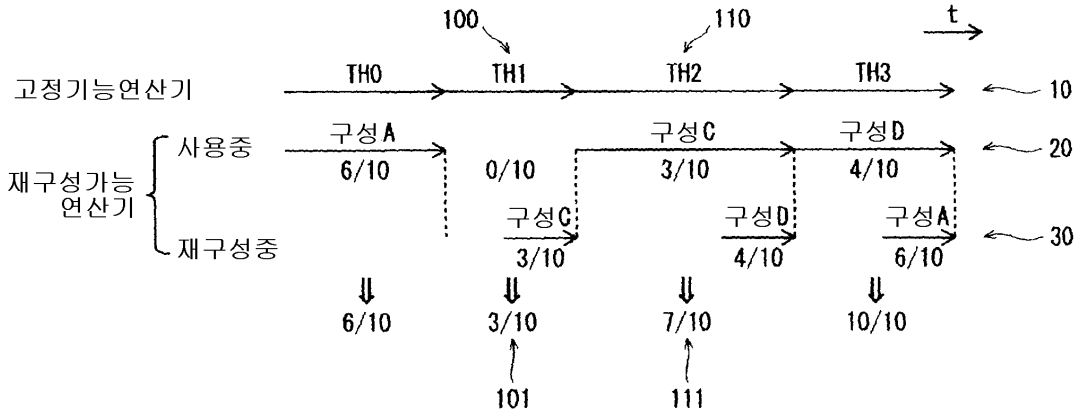
도면2

(a)

스레드명 (1411)	구성정보 (1412)	사용영역수(전체10) (1413)
TH0	구성A	6
TH1	—	0
TH2	구성C	3
TH3	구성D	4

(1410)

(b)

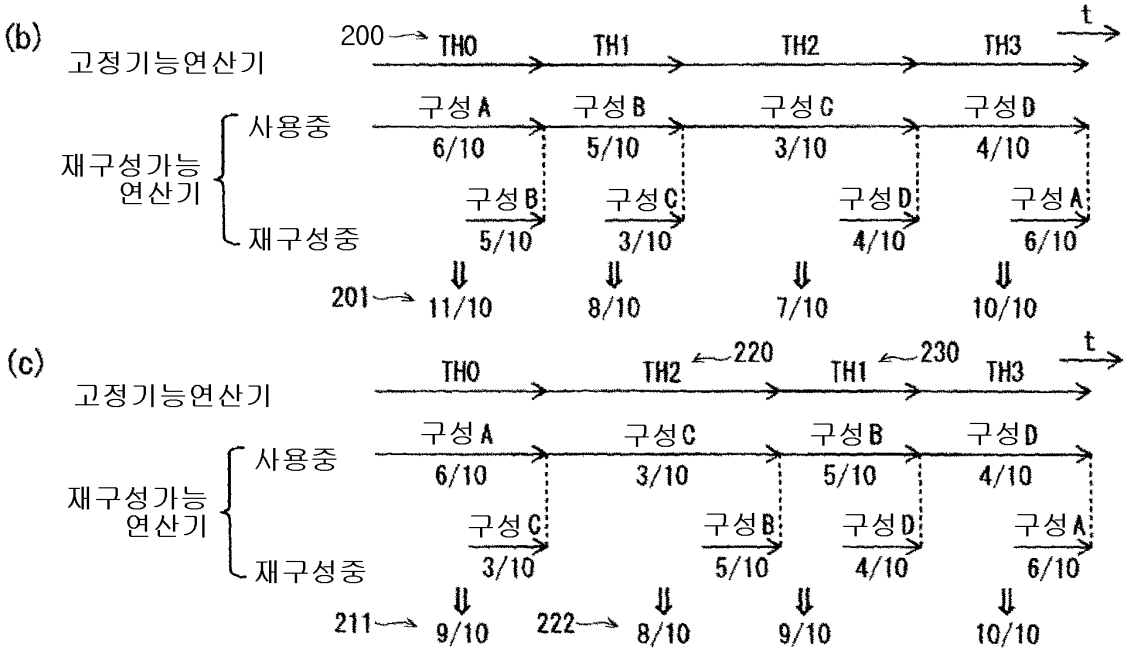


도면3

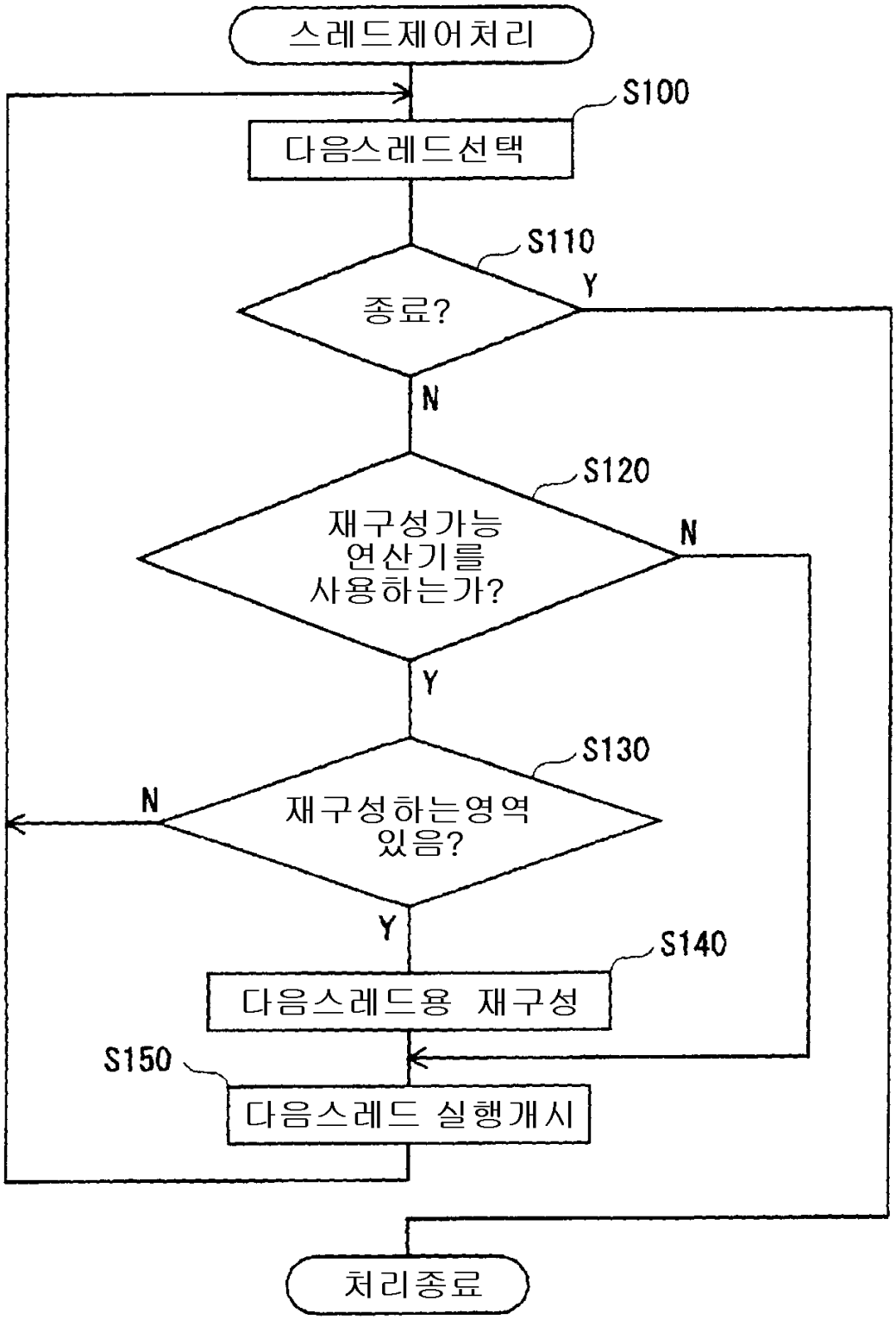
(a)

1411 스레드명	1412 구성정보	1413 사용영역수(전체10)
TH0	구성A	6
TH1	구성B	5
TH2	구성C	3
TH3	구성D	4

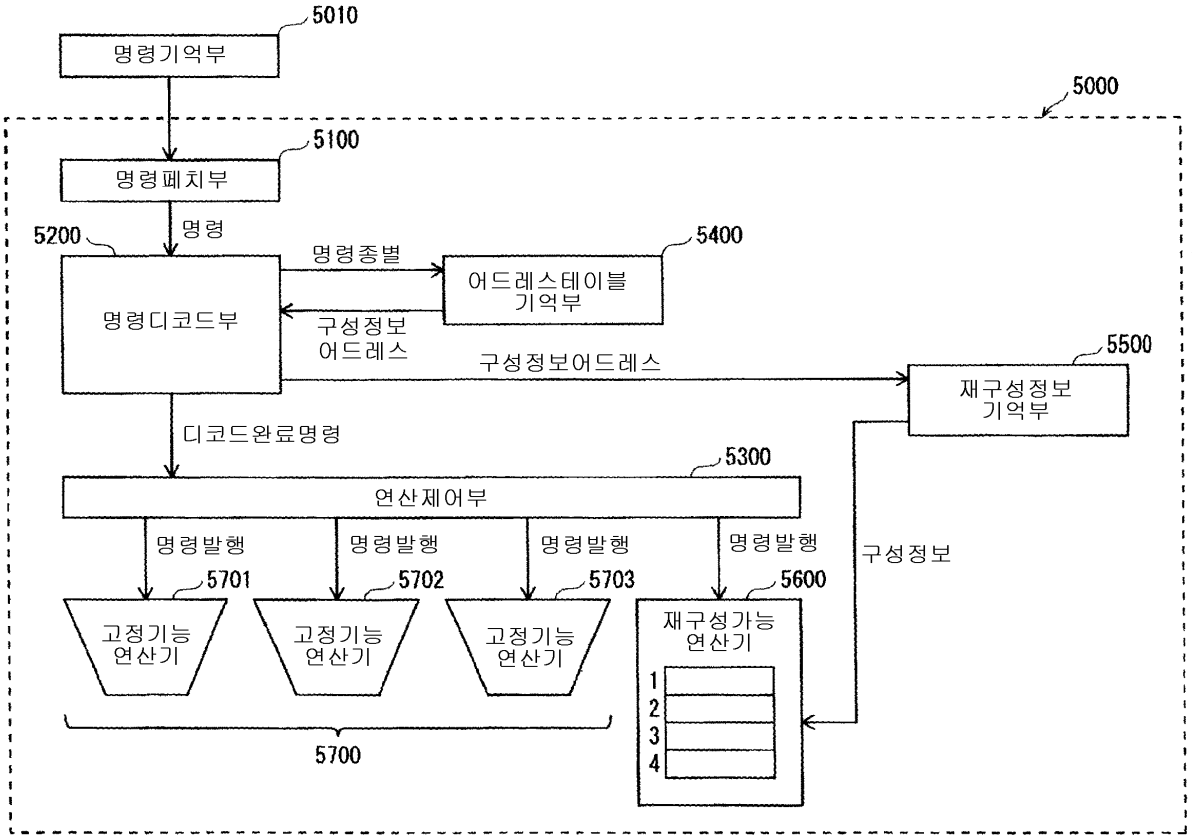
1420, 1421



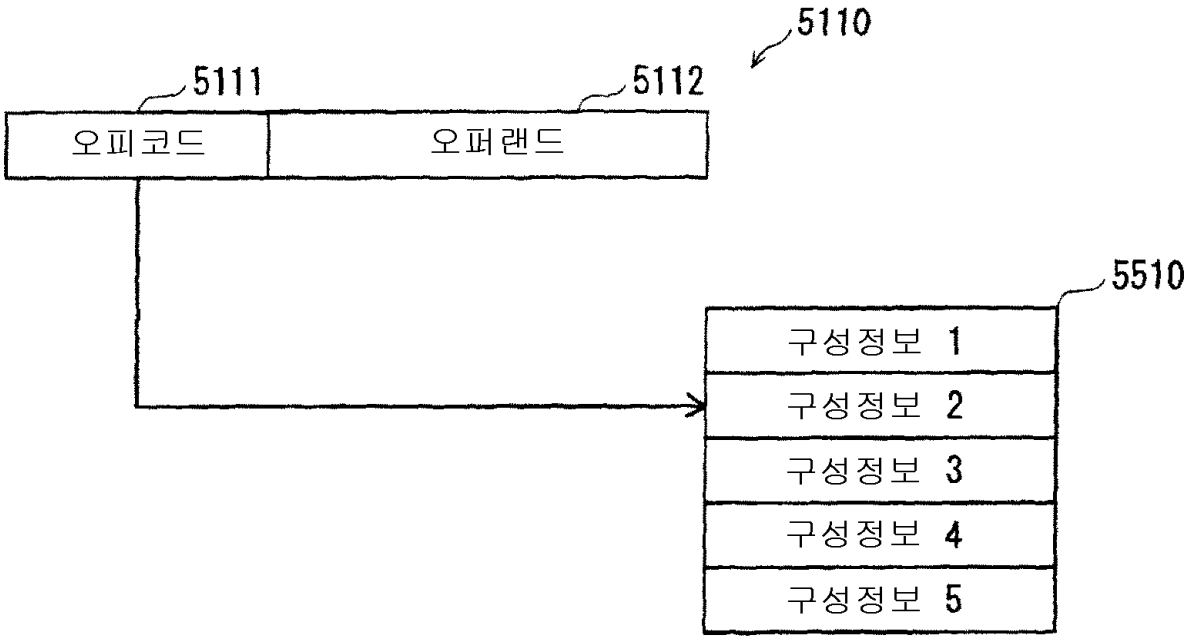
도면4


스레드제어처리
S100
다음스레드선택
S110
종료?
Y
N
S120
재구성가능
연산기를
사용하는가?
N
Y
S130
재구성하는영역
있음?
N
Y
S140
다음스레드용 재구성
S150
다음스레드 실행개시
처리종료

도면5


5010
명령기억부
5000
5100
명령페치부
명령
5200
명령디코드부
명령종별
구성정보
어드레스
5400
어드레스테이블
기억부
구성정보어드레스
5500
재구성정보
기억부
디코드완료명령
5300
연산제어부
명령발행
명령발행
명령발행
명령발행
구성정보
5701
5702
5703
5600
고정기능
연산기
고정기능
연산기
고정기능
연산기
재구성가능
연산기
1
2
3
4
5700

도면6


5110
5111
5112
오피코드
오퍼랜드
5510
구성정보 1
구성정보 2
구성정보 3
구성정보 4
구성정보 5

도면7

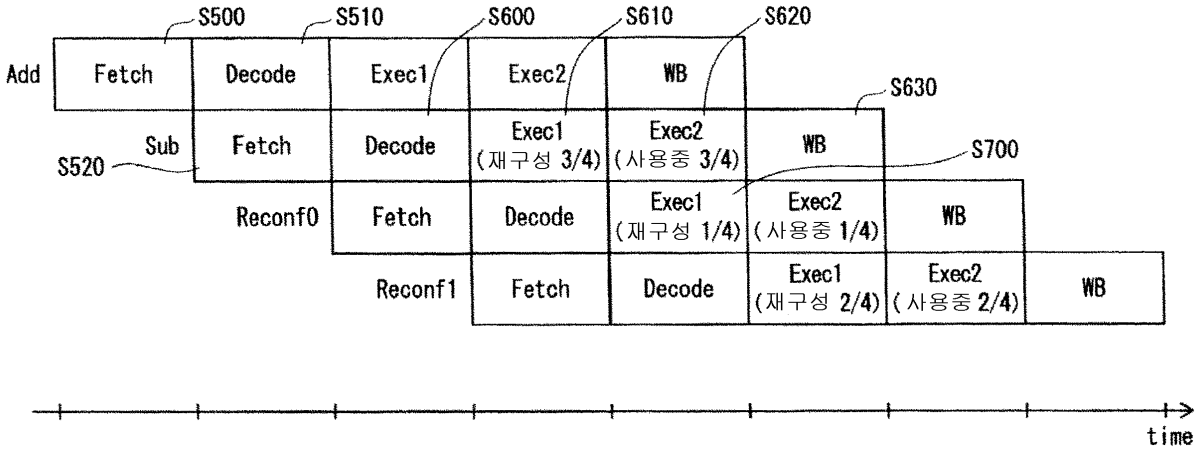
5411 오피코드종별	5412 어드레스	5413 사용영역수(전체4)
Sub	addr1	3
Mul	addr2	4
Dev	addr3	4
Reconf0	addr4	1
Reconf1	addr5	2

5410

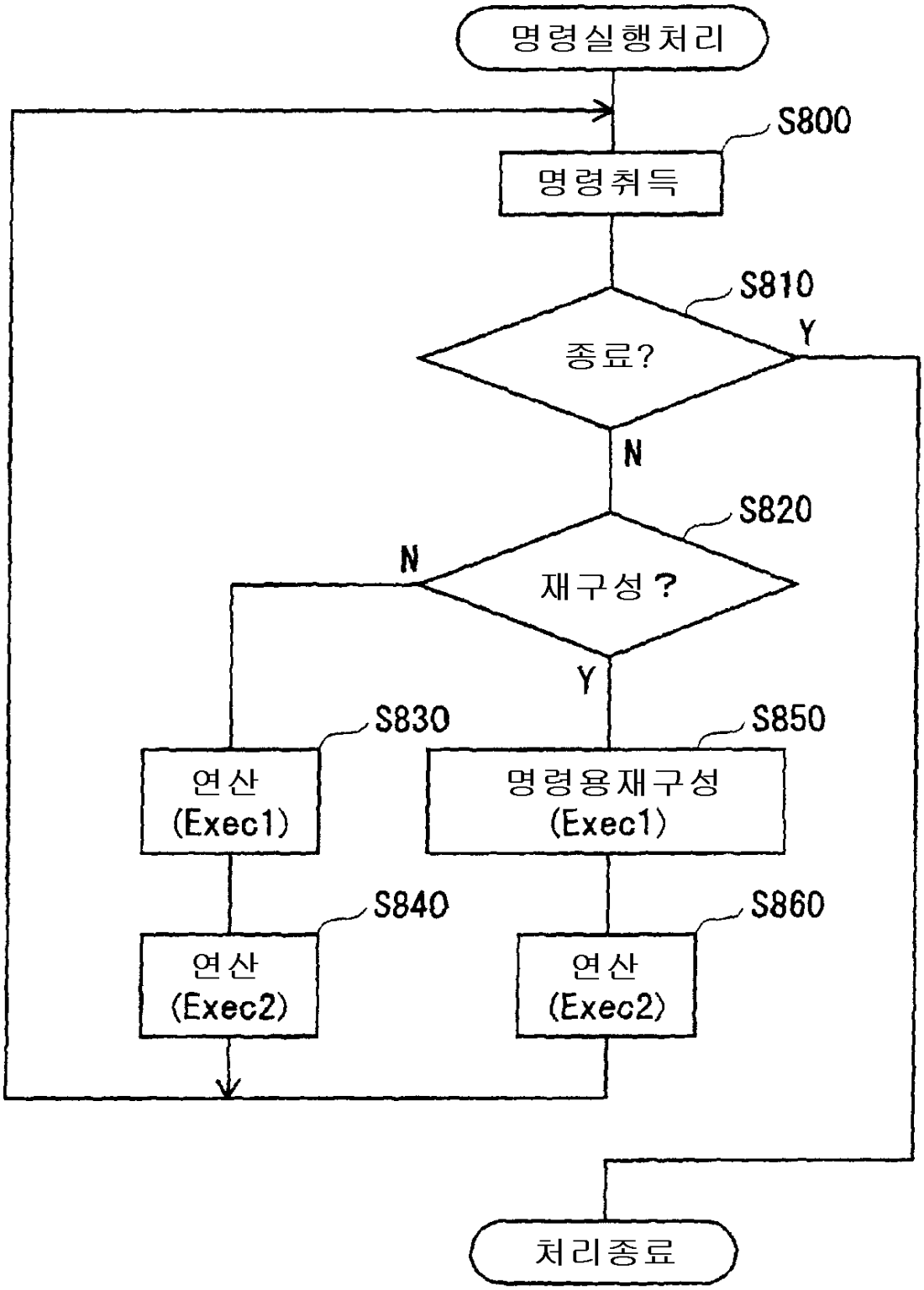
도면8

5111	5112
Add	r0, r1, r2
Sub	r3, r1, r3
Reconf0	r2, r0, 0xfe
Reconf1	r3, r1, r3

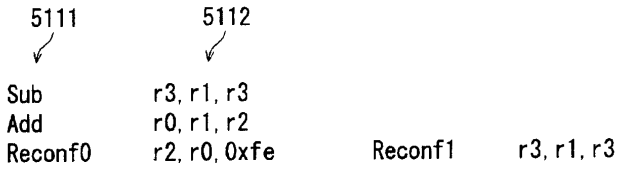
도면9



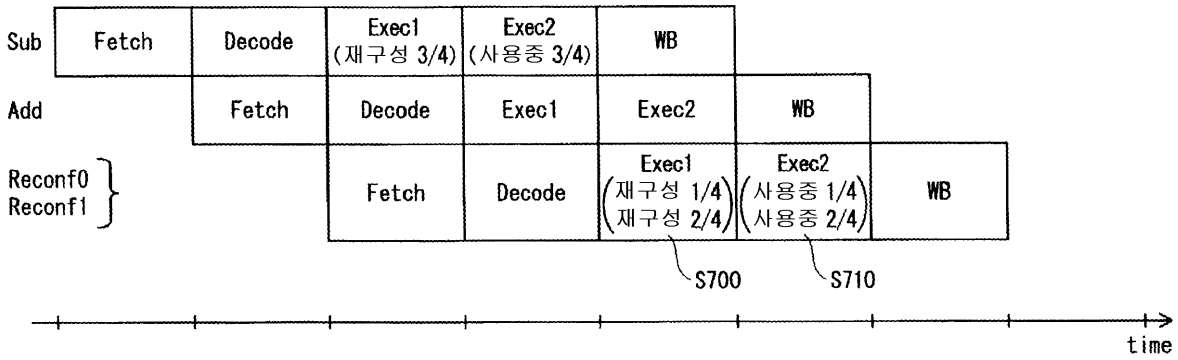
도면10


명령실행처리
S800
명령취득
S810
종료?
Y
N
S820
재구성?
N
Y
S830
연산
(Exec1)
S850
명령용재구성
(Exec1)
S840
연산
(Exec2)
S860
연산
(Exec2)
처리종료

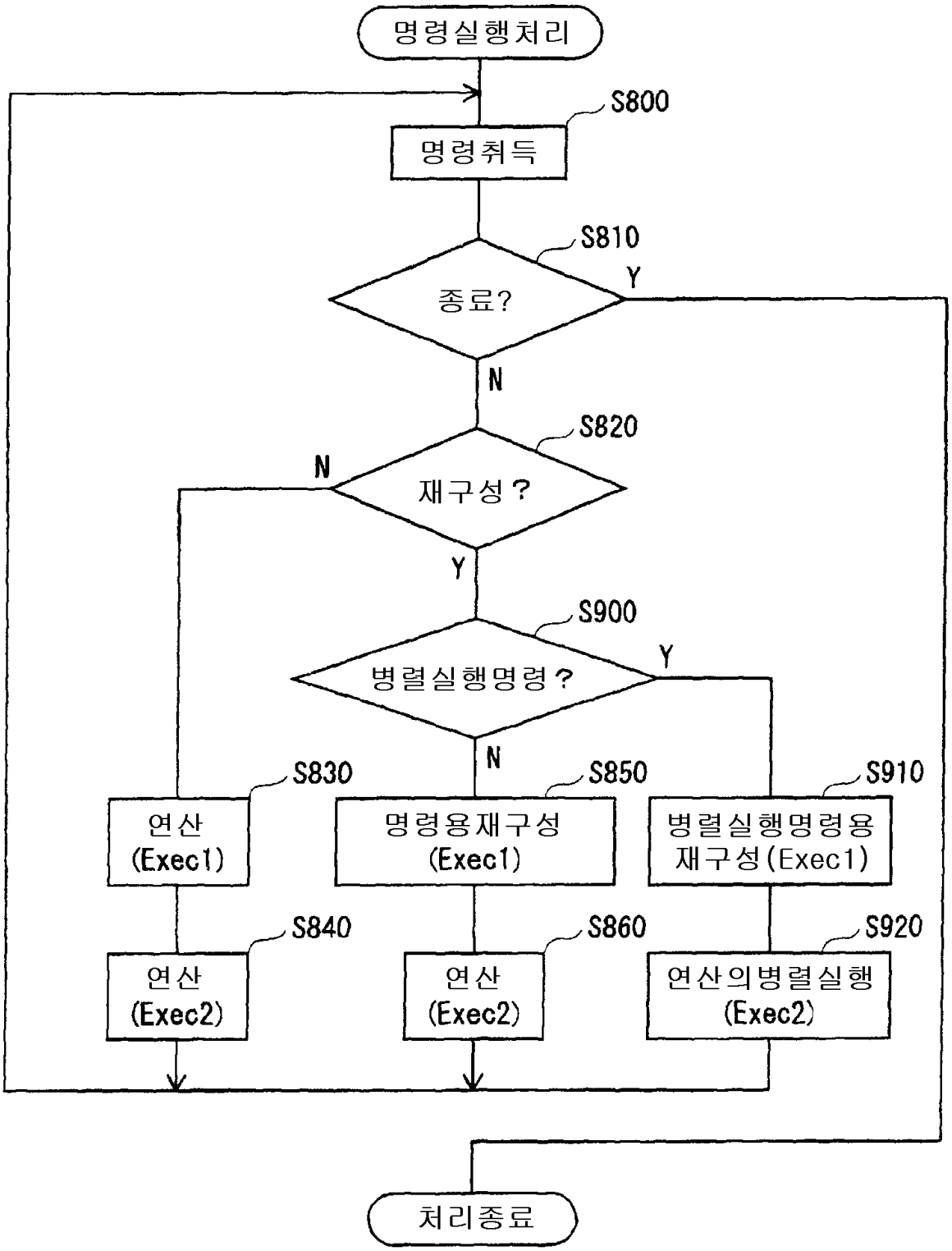
도면11


5111
5112
Sub r3, r1, r3
Add r0, r1, r2
Reconf0 r2, r0, 0xfe
Reconf1 r3, r1, r3

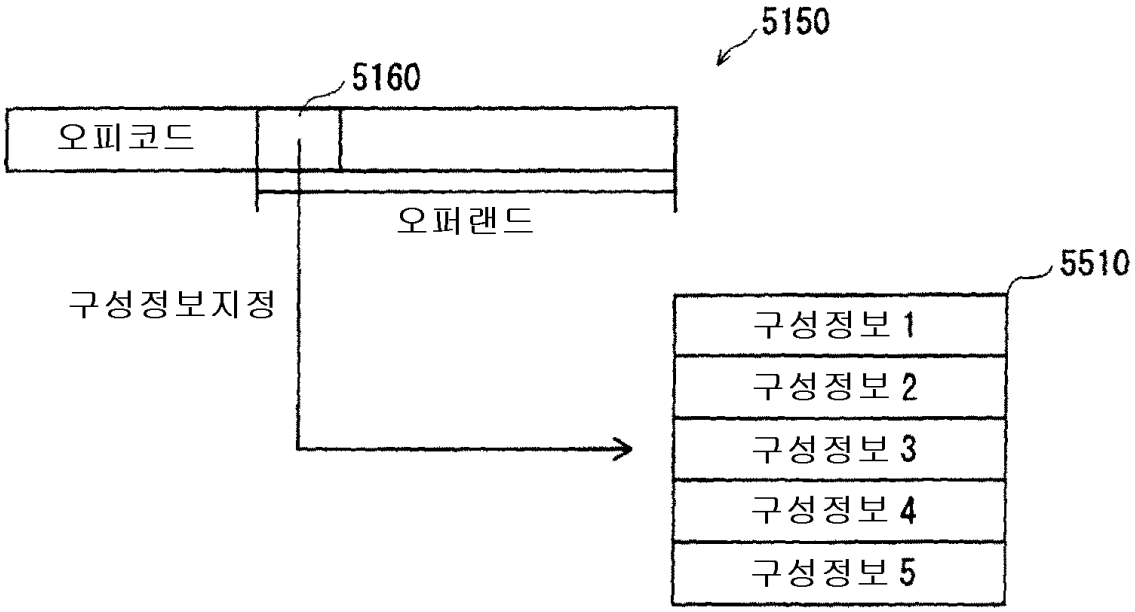
도면12


Sub
Fetch
Decode
Exec1 (재구성 3/4)
Exec2 (사용중 3/4)
WB
Add
Fetch
Decode
Exec1
Exec2
WB
Reconf0
Reconf1
Fetch
Decode
Exec1 (재구성 1/4 재구성 2/4)
Exec2 (사용중 1/4 사용중 2/4)
WB
S700
S710
time

도면13


명령실행처리
S800
명령취득
S810
종료?
Y
N
S820
재구성?
N
Y
S900
병렬실행명령?
Y
N
S830
연산
(Exec1)
S850
명령용재구성
(Exec1)
S910
병렬실행명령용
재구성(Exec1)
S840
연산
(Exec2)
S860
연산
(Exec2)
S920
연산의병렬실행
(Exec2)
처리종료

도면14


5150
5160
오피코드
오퍼랜드
구성정보지정
5510
구성정보 1
구성정보 2
구성정보 3
구성정보 4
구성정보 5