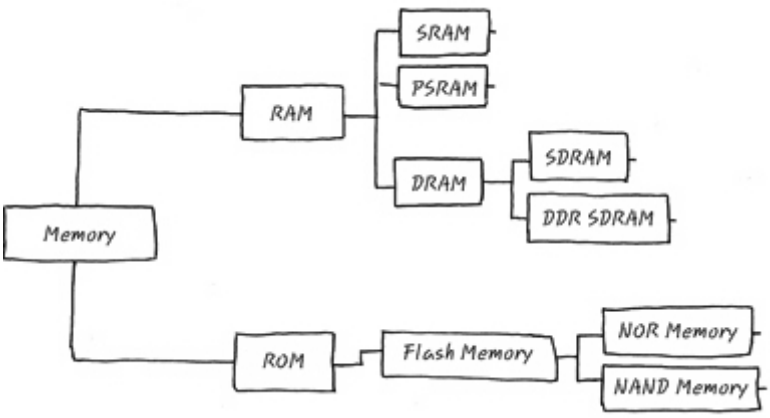
친절한 임베디드 시스템 개발자 되기 강좌 : Memory의 선정과 XIP

Memory의 선정과 XIP
Embedded System에서 Memory의 선정은 기본적인 System 구성과 Performance에 가장 큰 영향력을 행사합니다. 또한 Memory 종류의 선정을 어떻게 했느냐에 따라 Memory Map을 어떻게 구성할 것이냐의 Design Gudie가 달라지지요. Hardware 구성도 엄청 많이 달라지고, 가격에도 많은 영향을 미칩니다. 아.. 미칩니다.

잡설은 그만 하고, 우선은 우리 Embedded System에서 가장 많이 애용되고 있는 구성을 알아보기 위해서 Memory의 종류에 대해서 다시 한번 고찰해 보시도록 하시는 건 어떨지요.



뭐 엄청난 Memory종류를 모두 고려한다면야 좋겠지만, 너무 많은 건 우리 공부에도 별로 도움이 되지 않으니 이것들 이외의 것들은 나중에 알아보든가 일단은 모르는 척하는 것이 상책이에요. ㅋㅋ.
우선은 메모리는 크게 RAM과 ROM으로 나눌 수 있습니다. 보통 RAM을 휘발성, ROM을 비 휘발성이라고 부르는 데 그 이유는 알코올이 휘발성인 것처럼 휘발성 물질은 가만히 냅두면 증발되어 날아가 버린다는 뜻이 있죠. 마찬가지로 RAM은 전원을 나가면 RAM에 들어 있던 Data도 같이 날아가 버리고요, ROM은 전원이 나가도 Data가 계속 저장 되어 있는 성질의 메모리 입니다.

그러니까, 예전에는 RAM에는 Data가, ROM에는 Code가 들어가는 것이 통상적인 예였었는데요, 이제 조금은 concept을 달리해야 할 때가 온 것 같아요.

일단은 XIP (Execute In Place)라는 개념에 대해서 소개를 해야 하겠네요. XIP는 메모리 상에서 직접 program/ code를 실행 할 수 있는 기술을 말하며, 그 기본조건으로는 Random Access가 가능해야 합니다. 다시 말해, Byte/ Word/ 등의 크기를 직접 Access가 가능해야 한다는 뜻이며, 모든 RAM은 이런 요건을 충족합니다. 그러니까 RAM에 program/ Execution image를 올리기만 하면 실행 가능하다는 말씀입니다.

기왕 메모리종류를 그려보았으니, 뭐 하나, 하나 짚어 가면서 볼까요?

제일 비싼 넘은?

일단 RAM을 되짚어 보면, SRAM (Static RAM)은 제일 비싼 RAM입니다. 왜냐, 아무런 전기적인 control 없이 그저 우리가 생각하는 순수하게 Random Access가 가능한 휘발성 메모리 인 것입니다. 그냥 하릴없이 RAM이지요. 비싸요.

제일 싼 RAM은 어떤 것일까요? DRAM (Dynamic RAM)입니다. 왜 DRAM이 싼가? 하면 DRAM은 한번 Memory Cell에 뭔가를 기억 시킨다고 끝나는 게 아니에요. 시간이 지나면 슬슬슬 슬그머니 Charging되었던 전하가 빠져나가서 Data를 유실하게 되고요, 같은 주소 즉, 같은 자리를 계속 읽으면 Data가 유실 되어요. 전원을 끄지 않아도 점점 기억을 잃어가는 내 머리 속의 지우개이지요. 그래서 DRAM은 일정 시간 마다 다시 전하를 충전해 줘야 자기가 갖고 있던 Data를 잊어버리지 않고 계속 간직 할 수 있습니다. 당연하게도 DRAM에는 이런 charging을 위한 precharge circuit을 가지고 있어야 하는 거죠. 그러나, SRAM에 비해 회로가 단순하고 집적도가 높습니다. 그러니까 부지런히 charging을 해줘야 하는 대신에 같은 가격이라면 SRAM보다 더 큰 SDRAM을 살 수 있는 셈이죠. 이런 charging해주는 부분도 사용자가 PL172라는 규격을 통하여 DRAM control 해 줘야 하는 사용자의 수고로움도 있어요.

불편한 넘이 싼 법이죠.

자, 그러면 중간에 낀 PSRAM은 뭐냐? PSRAM은 Pseudo SRAM으로서 가짜 SRAM이에요. 이 녀석은 DRAM과 SRAM의 장점을 본 떠서 만든 RAM인데, 어떤 녀석이냐 하면, SRAM처럼 보이는 DRAM입니다. 이 녀석은 contorl을 해 줄 필요 없는 DRAM인데, Hardware적으로 precharge를 해주는 회로가 DRAM에 딸려서 나옵니다. 그러니까 알아서 DRAM charging control을 해주니까 사용자 입장에서는 마치 SRAM처럼 쓸 수 있는 게죠.

결국 같은 크기라면 가격이 SRAM > PSRAM > DRAM 인 셈입니다.

뭐 곁다리로 DRAM중 SDRAM과 DDR SDRAM을 을 comment한다면, SDRAM은 Synchronous DRAM으로서 동기식 DRAM이고 이 동기 - 박자 - 라는 건 System Bus와 쿵짝 동기를 맞춘다는 뜻으로 거의 CPU의 동작에 맞추어 동작한다고 봐야 합니다. 박자를 맞춘다는 건 CPU가 보기에는 최대의 성능을 낼 수 있다는 뜻이죠. DDR SDRAM은 Double Data Rate로서 System Bus와 동기는 system bus의 clock으로 맞출 수 있는데 이 bus clock의 rising edge와 falling edge에서 모두 동작을 하므로 Data가 두배로 빨리 전송될 수 있는 Memory를 말합니다.

참고로, Asynchronous RAM은 Address line의 input을 받은 후 일정 시간 동안 (Wait state) 내부 동작을 거친 후 그 주소의 값을 output으로 내 놓는 구조인데, 이는 항상 wait state만큼의 버리는 시간이 존재하게 되니 Synchronous보다 Latency가 많다고 봐야겠죠.

자, 열심히 RAM을 봤으니 이번엔 ROM을 한번 봅시다. Embedded System에서는 쓰고 "지울" 수 있는 Flash Memory가 ROM으로 가장 많이 애용됩니다. Flash memory라 함은 ROM이긴 하지만 특별한 control 과정을 거치면 지울 수도 있고 그 위에 다시 쓸 수 도 있습니다. 그러니까 언제라도 Burning (Programming)가능한 Flash는 Embedded 개발 환경에서 엄청나게 인기가 있을 수 밖에 없습니다. Upgrade에도 유리하고 심지어 지울 수 있는 기능 때문에 File system storage로도 사용됩니다.

둘 중에 어느 게 더 비쌀까요? 같은 크기라면 NOR가 NAND보다 더 비쌉니다.

NOR > NAND

일단 NOR와 NAND의 차이는 NOR는 Cell이 병렬로 연결되어 있으며 병렬로 연결되어 있다 보니, Address Line과 Data Line을 모두 가질 수 있으며 RAM처럼 byte단위로 Random Access가 가능하게 합니다. 어랏? 여기서 Random Access 얘기가 나오니 생각나는 얘기가 있네요. 그렇습니다. NOR는 XIP를 지원합니다. XIP를 지원한다는 얘기는 Software을 직접 실행할 수 있다는 얘기지요.

혹자는 Software를 메모리에 로드하여 실행하는 대신 플래시에서 직접 수행하는 기술을 XIP(eXecution-In-Place) 기술이라고 하여, Flash 소자에 그 용례를 국한하기도 합니다만, 저는 모든 메모리 소자들에 XIP라는 용

어를 적용하고 싶네요.

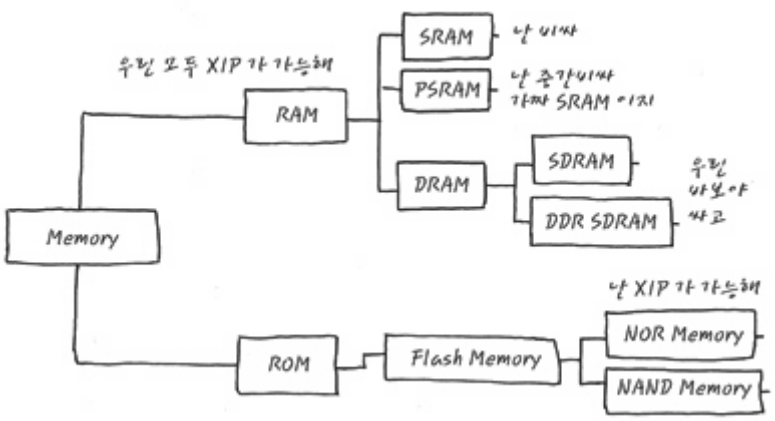
결국에 저는 Word단위의 Access가 가능하여, Software를 Execution(실행)할 수 있는 것을

XIP라 정의하겠습니다.

왜 이런 얘기를 하느냐? 그럼 NAND는 XIP가 안되느냐? 그렇습니다. NAND는 XIP를 지원하지 않습니다. 왜냐, NAND는 기본적으로 Cell이 직렬로 연결되어 있으며, 작은 단위로는 읽을 수 없고, 한번 읽을 때 1개 page단위로만 읽을 수 있습니다. Small page를 지원하는 NAND는 512 byte크기로만 읽을 수 있고, Large page을 지원하는 NAND는 2KB씩 한번에 읽을 수 있겠습니다.

NOR형은 쓰기와 지우기는 느리지만 읽기가 빠르다는 특성이 있지만 대용량 데이터를 저장하는 데에는 한계가 있어요. 반면 NAND형은 읽기는 느리지만 쓰기와 지우기가 빠릅니다요. 예예. 또 대용량 저장이 가능하다는 장점이 있지요. 읽기가 빠른 것을 선택할 것이냐, 쓰기와 지우기가 빠른 것을 선택할 것이냐에 따라 어떤 플래시메모리를 사용할 것이냐는 운명이 결정되는 것이지요. 크흑.

그러다 보니 참으로 특이한 정의들이 나오기 시작했는데 NOR는 코드저장용 NAND는 데이터 저장용이라는 말이 막 횡횡하기 시작했는데 뭐 이런 XIP의 연유 때문에 그런 말이 나온 듯 합니다. 후후.



여하튼 이런 Memory의 종류를 보니, 보통 Flash Memory 업체에서 RAM에 까지 손을 뻗쳐서 Flash memory와 RAM을 한꺼번에 한 칩에 집적한 선물 포장세트를 만들어 내기에 이르렀으니 이를 일컬어 MCP (Multi chip package)라고 부릅니다.
그러면, MCP를 만드는데 여러 가지 조합이 있을 텐데, 어떻게 조합을 하면 좋을까요? 일단은 XIP가 가능한 형태로 그 system을 구성해야 하는 것이 정답입니다. 그래야 Software를 실행, 동작시킬 수 있겠죠.

일단 예전에는 NOR + PSRAM의 구성이 유행을 했었습니다. NOR는 XIP가 가능하니까, NOR에 Code를 넣고, PSRAM에 Data를 넣는 형태를 취했었죠. 이런 system은 Code의 크기가 크지 않고, Data의 크기도 그리 크지 않은 system에 적합했으니까 예전에 많이 유행했습니다. 또한 Flash에서 직접 code를 실행할 수 있고, Flash를 마치 RAM처럼 Access 가능하기 때문에 user의 개발편의성이 아~주 좋았습니다.

그럼 요즘은 어떤게 유행 할까요? 요즘은 대형화 추세에 발맞추어, NAND + (DDR) SDRAM의 형태에 발맞춘 MCP가 유행을 하고 있지요. NAND는 단지 코드 저장용이고 이런 코드를 SDRAM에 복사를 하여 RAM에서 직접 XIP를 하는 형식으로 발전해 가고 있습니다. 와랏.

이런 추세는 대형화 대용량 추세에는 걸맞지만 이런 시스템을 전체적으로 관리해 줘야 하는

user의 개발 편의성이 아~주 복잡해 지긴 했습니다.

Flash는 NAND건 NOR건 간에 block 단위로만 지울 수 있습니다. NAND의 경우에는 32개 또는 64개 page로 이루어져 있고, NOR도 128KB단위의 block 단위를 갖습니다. 특이한 경우는 NAND나 NOR의 erase는 0으로 지우는 것이 아니라, charging을 하는 경우라서 지우고 난 block은 모두 1로 채워져 있어서 16진수로 보면 0xFF로 가득 차 있습니다. 하하. 이거 처음에 모르면 참으로 당황스럽다니까요.

by 히언 | 2009/05/31 21:26 | 하드웨어꼴라쥬 | 트랙백 | 핑백(2) | 덧글(2)