EDIF 2 0 0 - 초보 사용자를 위한 기존 EDIF 파일의 가져오기 변환 안내

소개

EDA 물리적 레이아웃 툴은 GDSII 데이터에 대한 스트림 입력 및 스트림 출력 플로우를 제공합니다. 이 데이터는 일반적으로 IC 제조를 위해 제공하는 최종 표준 데이터 형식으로 이해됩니다. 안타깝게도 설계 주기의 앞 부분에서 데이터는 훨씬 더 주관적입니다. 스키매틱 캡처 툴과 에디터는 데이터의 표현 및 활용 방식에 따라 매우 다양합니다. 또한 업계에서 사용할 수 있는 시뮬레이션 엔진에 대해 하나의 넷리스트 형식만 있는 것이 아니므로, 작성하는 넷리스트는 훨씬 더 다양할 수 있습니다. 툴을 호환하여 사용할 경우, 스키매틱 및 라이브러리 데이터 형식을 벤더 간에 마이그레이션하는 방법은 매우 중요합니다. 어느 벤더의 플랫폼에서 다른 벤더의 플랫폼으로 완전히 마이그레이션하는 경우, 이 문제는 더욱 어려워질 수 있습니다. 본 애플리케이션 노트에서 Gateway 스키매틱 에디터를 사용하여 EDIF 2 0 0 형식을 최대한 활용하는 방법에 대해 몇 가지 실용적인 통찰력을 제시합니다.

변환 시나리오

어떤 벤더의 스키매틱 라이브러리 데이터를 마이그레이션할 계획이 있는 사용자라면 무엇이 관련되는지 이해하는 것이 중요합니다. EDIF 표준은 대부분의 스키매틱 정보를 제공하지만, 전체적으로 표준을 지원하는지는 개별 EDA 벤더에 달려 있습니다. 모든 벤더가 가져오기 및 내보내기 루틴에서 표준의 모든 부분을 지원하는 것은 아닙니다. 전체 표준을 지원하는 벤더의 경우에도, 벤더별로 고려해야 하는 몇 가지 제한 사항이 있습니다. 주요 제한 사항은 가져온 스키매틱에 개별 요소에 대해 관련된 넷리스트 정보가 없다는 것입니다. 넷리스트 구문은 시뮬레이터에만 특정되며, EDIF 표준은 요소 간의 연결을 제외하고 넷리스트에 대한 데이터를 제공하지 않습니다.

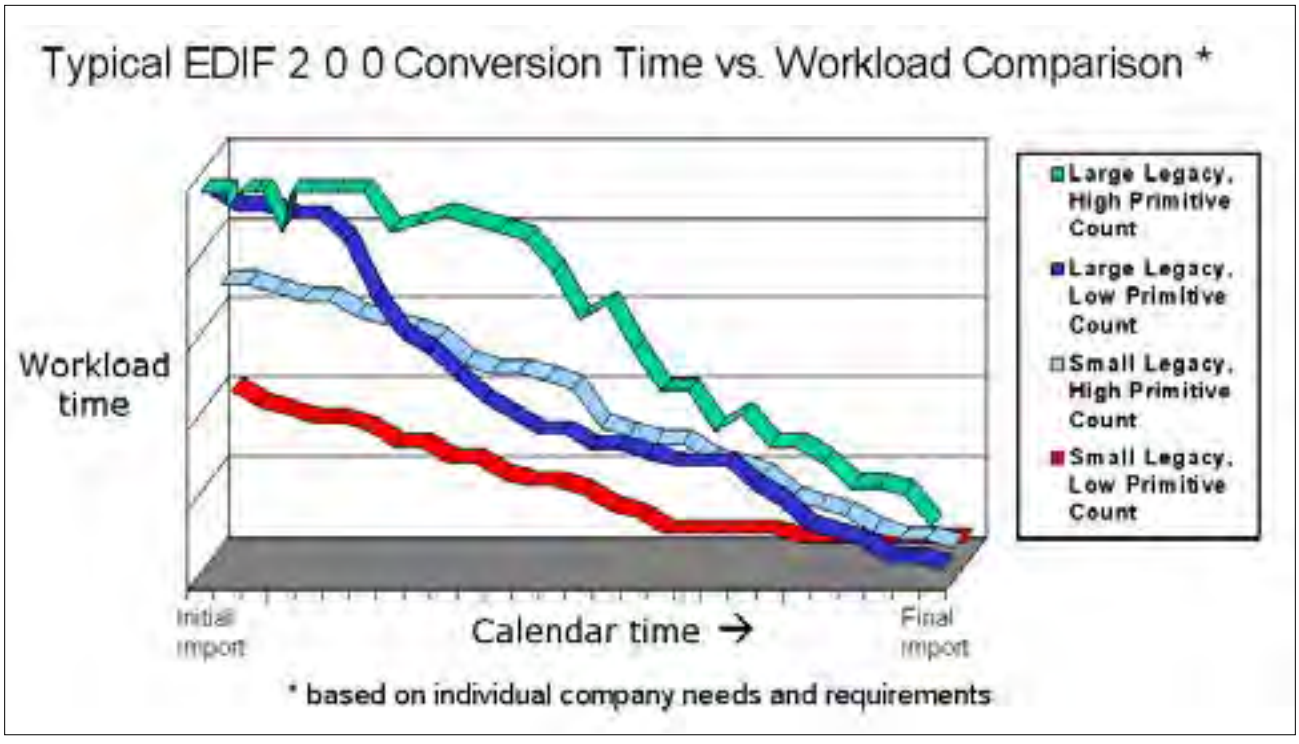

그림 1.

그러나 각 인스턴스에 대한 특정 소자 동작(설계 의도)은 여전히 고려하지 않습니다. 이는 EDIF를 사용할 때 가장 중요한 개념일 수 있습니다. 예를 들어, 어떤 소자가 가져오기를 수행하여 4핀 트랜지스터처럼 보이더라도, 서브 회로에서 또는 부착된 매크로모델 설명을 사용하는 기호가 아닌 기본 소자임을 확인할 수 있는 유일한 사람은 원본 상태에서의 설계에 익숙한 사람입니다. 가져온 요소의 설계 의도를 알 수 없거나 설명할 수 없는 경우, 스키매틱이 제대로 표시되지 않습니다. 비유하자면, 설계를 레이아웃할 수 있지만 배치 및 배선은 다른 사람에게 맡길 수 있습니다. 그러나 테이프 아웃 전에 항상 배치 및 배선을 고려해야 합니다. 여기서 EDIF는 가져오기를 수행하는 툴이지만, 각 셀의 유형이 충실하게 표현되도록 설계 의도를 가지고 가이드해야 합니다.

그림 1은 서로 다른 기존의 설계 상황을 나타내는 네 가지 시나리오를 나타냅니다. EDIF 가져오기 및 변환은 완전히 다른 유형입니다. 가져오기는 파일 및 Gateway 자동 라이브러리의 가져오기와 EDIF 파일에 포함된 셀 생성을 의미합니다. 변환은 가져온 데이터를 회로로 최대한 충실하게 시뮬레이션하고 나타낼 수 있는 유용한 넷리스트로 변환하는 과정입니다. 그림 1은 가져오기 및 변환에 대한 설명입니다.

그래프의 x축은 "X" 벤더에서 Gateway로 설계 데이터베이스를 가져와서 변환하게 되는 시작부터 끝까지의 시간을 나타냅니다. Y축은 작업 시간 또는 변환에 소요된 작업 시간의 백분율을 나타냅니다. 그래프를 보면 시간이 지남에 따라 데이터를 가져오고 변환하고 확인하는 데 소요되는 시간이 서로 다른 속도로 감소한다는 것을 알 수 있습니다. 감소율은 가져오고 변환할 기존 데이터의 크기와 설계 데이터베이스의 기본 요소 수에 비례합니다. 기본 소자는 하위 회로 또는 하위 스키매틱으로 표현되는 것이 아니라 진정한 SPICE 기본 요소로 기능하는 소자입니다.

기존 데이터를 Gateway로 처음 가져올 때, 모든 요소를 가져와야 합니다. 향후의 가져오기를 위해, Gateway는 모든 항목을 다시 가져오거나 이전에 가져온 셀을 건너뛸 수 있는 옵션을 제공합니다. 대부분의 사용자는 다음과 같은 이유로 데이터베이스에 이미 있는 셀을 건너뛰도록 선택합니다.

1) 시간이 절약됩니다.

2) 셀을 가져온 경우, 넷리스트를 위해 셀을 변환하는 데 시간이 걸렸을 수 있으며 변경 사항을 잃고 싶지 않습니다.

각 설계를 가져오면, 점점 더 많은 기존 설계가 Gateway 내부에 구축됩니다. 또한 더 많은 설계를 가져올수록 가져올 새 셀이 줄어들게 됩니다. 이는 대부분의 설계가 이미 가져와서 Gateway 라이브러리에 저장된 기존 설계를 재사용하기 때문입니다.

그림 1을 보면 시간 경과에 따라 변환하는 데 가장 많은 작업이 필요한 데이터베이스는 많은 라이브러리/셀과 수많은 기본 소자가 있는 데이터베이스입니다. 이 경우, 초기 작업은 기본 소자가 적절히 변환되는지 확인하는 데 집중됩니다. 비록 현실에는 한정된 수의 진정한 기본 소자가 있지만, 기본 소자에 대한 변형이 많이 있을 수 있습니다. 핀이 4개인 pmos 기호를 예로 들 수 있습니다. Mosfet 요소라고도 하는 "M" 요소인 pmos 기본 소자는 SPICE에 하나만 있습니다. 그러나 특정 모델, 속성 및 값이 연관된 특정 pmos 셀로 사용한 경우, 사용자 데이터베이스는 하나의 pmos 소자 또는 수십 개의 pmos 소자를 포함할 수 있습니다. 이러한 상황은 수많은 기존 요소와 수많은 기본 요소가 있습니다. 그림 1에서 녹색 선은 이를 나타냅니다.

많은 셀과 적은 기본 요소를 포함하는 기존 설계는 그림 1에서 진한 청색 선으로 표시됩니다. 예를 들어, 동일한 pmos 및 nmos 소자를 사용하는 많은 셀이 있습니다. 이러한 pmos 및 nmos 소자는 처음 가져올 때 발견할 가능성이 높으므로, 한 번 변환한 후 완료해야 합니다. 그런 다음 이를 사용하여 가져온 설계는 더 이상 수정하지 않아도 됩니다. 이와 대조적으로, 그림 1의 적색 선은 기본 요소가 거의 없는 소규모 기존 설계를 나타냅니다. 곡선을 보면 처음 몇 개의 가져오기를 사용하여 기존 요소를 변환하고, 그 이후에는 최소한의 노력으로 가져오기 및 변환을 거의 한 번에 한다는 것을 알 수 있습니다.

결론

EDIF 가져오기 툴을 사용하여 벤더의 기존 제품을 Gateway로 가져오려면 원본 설계를 숙지하는 것이 중요합니다. 설계 의도와 함께 라이브러리의 기존 크기 및 셀 유형 (기본 대 비기본)은 성공적인 변환에 필요한 시간과 노력을 결정할 때 고려해야 할 중요한 요소입니다.