AXIS Tool 을 활용한 신호처리개발에 관한 연구

김도훈*, 정기현**
*아주대학교 전자공학과, 삼성탈레스
** 아주대학교 전자공학과
e-mail : cogito@ajou.ac.kr *, khchung@ajou.ac.kr **

A Study on AXIS Tool for Radar signal Processing

Do-Hoon Kim*, Kihyun Chung **
Dept. of Electronic Engineering, Ajou University,

요 약

레이더 시스템에서 신호처리에 대한 연산 량 및 데이터의 전송 용량은 시스템 개발 시, 성능과 구성을 결정하는 주요 요소다. 이런 요소에 대한 정확한 사전 예측은 시스템 전체 성능 및 개발 기간과 연구비용에 영향 끼친다. 레이더 신호처리에 대한 정확한 연산 량 및 데이터 전송 량의 예측은 개발 전문가의 경험과 COTS 보드의 성능으로 결정된다. 이런 예측을 보다 정확하게, 보편적으로 하기 위해서 시뮬레이션을 이용하는데, 그 중 하나인 AXIS Development Tool 은 개발자에게 편리성과 효율성을 제공한다. 이 시뮬레이션 기능은 개발 보드와의 동일 환경을 제공함으로 개발의 시간을 단축시키고, 사용자의 편리한 GUI 환경을 제공함으로 개발의 유연성을 제공해 개발 성능에 대한 예측이 정확하여 안정적 개발을 보장한다.

1. 서론

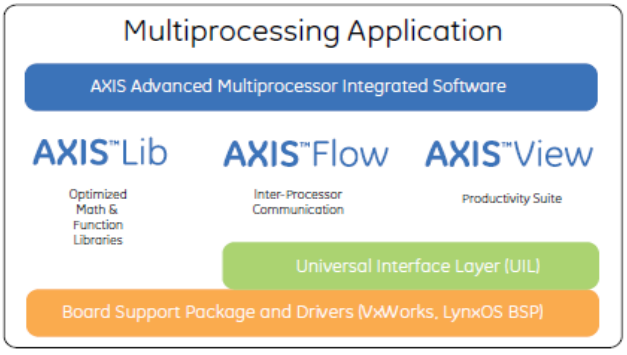
최근 많이 사용되는 COTS(Commercial Off-The-Shelf)보드는 고성능의 멀티 프로세서와 메모리의 발전에 의해서 성능이 급속도로 향상 되었다. 이런 멀티 프로세서는 이론상 복잡한 신호처리를 멀티 프로세서의 할당을 통하여 빠른 결과를 수집할 수 있는 구조가 되었다. 하지만, 이런 멀티 프로세서 구조에서 관련 알고리즘과 제어 소프트웨어는 더욱 더 복잡한 구조로 변해가고, 이에 대한 추가적인 노력이 필요하다. 이를 보다 효율적으로 개발하기 위해선 이를 지원하는 개발 TOOL 사용이 필수적인데 본 연구에서는 AXIS TOOL 을 통해서 멀티 프로세서 상에서 프로그램 개발 방법론과 TOOL 의 효과적으로 사용하는 방법에 대해 제안한다.

2. 신호처리 개발의 절차

레이더 시스템 등의 복합 시스템을 설계하는 할 때, 개발 과정의 많은 실수는 알고리즘의 수행시간, 데이터 사이즈 및 시스템의 변경 등 많은 부분이 설계 시, 결정 되지 않은 상태에서 진행되면서 구현 단계에서 일어난다.

이런 일들은 실제 구현 단계에서 많은 어려움을 유발하게 된다. 시스템의 설계 변경에서 치명적인 요소 중 하나는 알고리즘의 연산량 및 Throughput 향상을 위해서 연산 보드의 추가와 인터페이스의 변동 등의 하드웨어 구조 변경 및 이에 따른 소프트웨어의 개발 및 개발 일정 지연 등의 변수가 발생한다. 이런 오류와 변경을 피하기 위해서 초기 시스템 설계 시, 기 경험 설계자의 기본 설계를 활용하고 알고리즘 연산 량을 초기 계산하여, 시뮬레이션을 통하여 예측한다. 이런 개발 과정에서 AXIS TOOL을 활용하여 어떻게 변경사항에서 편리하게 재 구성 하고 예측시간 분석을 하는지를 정확한 레이더 신호처리기 개발 과정의 예시를 통해 다음의 순서로 알아보겠다.

1. AXIS TOOL의 구성 및 특징
2. 레이더 신호처리기의 구성 및 특징
3. 신호처리 개발 시 AXIS 적용 방안



(그림 1) Axis 구성도[2]

2.1 AXIS TOOL 의 구성 및 특징

AXIS의 Tools은 GE-IP에서 만든 개발환경 도구로

(그림 1)과 같이 AXIS Lib, AXIS Flow 및 AXIS View 등의 3가지 개발 Tool로 구성되어 있다.

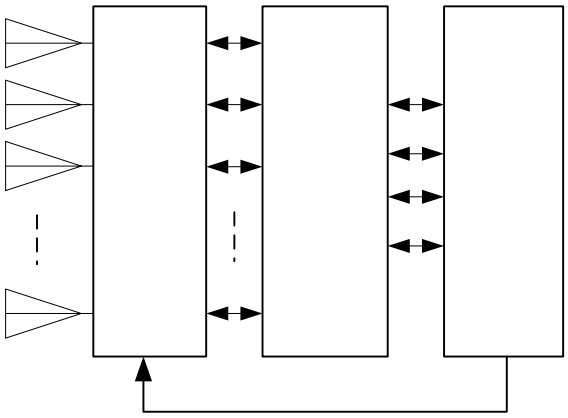
AXISView는 소프트웨어 개발자가 Multi-Process core 환경에서 GUI를 통한 편리한 개발 환경을 지원하는 suite 이다. AXISView는 하드웨어의 자원을 보는 RuntimeViewe와 Time Domain에서의 Event의 시간 흐름을 알 수 있는 Event View를 지원함으로써 자원의 사용을 실시간 적으로 그래픽컬하게 확인 할 수 있다.

AXISFlow는 시스템 설계에서 중요한 내부 I/O와 외부 I/O를 지원한다. AXISFlow의 라이브러리는 H/W에 대한 최적화 설정이 이미 라이브러리로 되어 있어 단순한 I/O에 대한 개발이 가능하다. 또한, AXISFlow는 AXISView의 도구를 통하여 Multi-Buffer, FIFO Message, I/O device등을 지원하고, P2P, multicast, M2M의 통신 채널 프로토콜을 보유하고 있어 multi-processing의 통신을 단순화 시켰다.

AXISLib은 신호처리 및 연산에 대한 전용 수학 라이브러리다. 기본 수식 라이브러리와 matrix, vector 연산을 제공하여 최적화한 라이브러리 이다.

2.2 레이더 신호처리기의 구성 및 특징

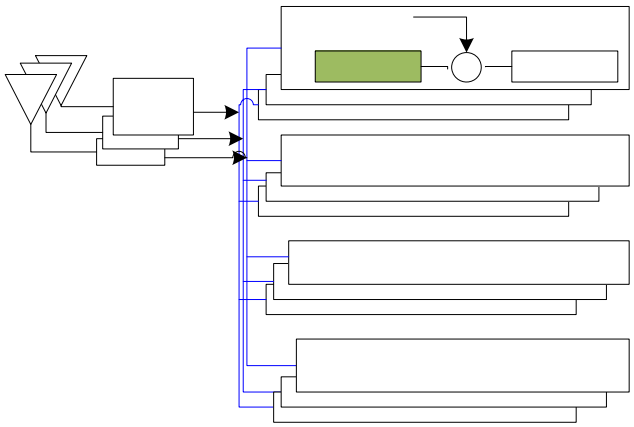
현대의 레이더는 안테나의 구성 소자의 발전과 연산처리 능력의 개선으로 다양한 외부 환경에서 높은 동작 성능을 나타내는 능동형 위상 배열 레이더 시스템으로 발전하고 있다. (그림 2)은 능동형 위상 배열 레이더 시스템의 구조를 나타낸다. 능동형 위상 배열 레이더는 다수의 안테나 배열 소자를 조합시켜서 배열 소자의 위상을 조정하여 전자적으로 빔을 수평과 수직 방향으로 조향시켜 사물의 위치를 탐지 및 추적하는 레이더 시스템이다[1].



(그림 2) 레이더 시스템 구조

각 안테나 배열 소자로부터 수신되는 외부신호는 고속 ADC 신호로 변환하여 레이더 신호처리기로 샘플링된 IQ 변환 신호가 입력된다.

능동 배열 레이더 신호처리기의 일반적 신호 처리 흐름은 (그림 3)과 같고, 여러 채널로부터 수신된 IQ 샘플링 신호는 빔 형성과정을 수행 후, 펄스 압축과 도플러 신호처리의 알고리즘을 수행한다. 펄스 압축 알고리즘과 도플러 처리 기술은 고속 퓨리에 변환(FFT)과 Hamming Window processing 및 Inverse FFT 를 사용하여 표적 신호를 추출한다. 이런 신호처리 과정은 많은 전송용량과 연산시간 및 메모리가 소모된다.



(그림 3) 레이더 신호처리 흐름

2.3 신호처리 개발 시 AXIS 적용 방안

위에서 간략하게 설명된 레이더 신호처리 흐름을 바탕으로 레이더 신호처리의 기존 설계방안에 대하여 알아보고 이와 함께 AXIS TOOL을 이용하여 레이더 신호처리 설계방안에 대하여 고찰하며, 기존설계와의 차이점을 살펴본다.

먼저, 레이더 신호 처리기에 대한 기존 설계 방안은 시스템 알고리즘 및 성능을 바탕으로 입력 샘플링 신호의 IQ량을 결정하고 이에 따른 알고리즘 연산 량 및 전송 용량에 대한 계산한다. 아래 <표 1>과 같이 주요 설계 지표를 기준으로 각 알고리즘의 수행성능 및 수행시간을 기술하여 보드의 적정 수량 및 가능성 여부를 확인한다.

<표 1> 주요 설계 지표 (예시)

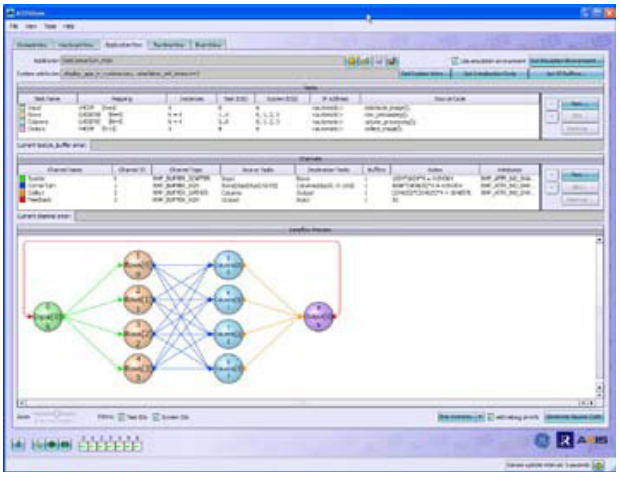
처리 알고리즘	시간 / 용량	비고
IQ 샘플링 개수	10000	PRI 별
FFT 연산속도	5usec	256 point
Window 연산속도	5usec	Ref. code
IFFT 연산속도	5usec	256 point
전송용량	1Gbps	10000*16bit/1Gbps

<표1>은 COTS 보드 제작 회사들의 매뉴얼과 전문가의 기술적 경험 등을 기반으로 작성된다. 이런 예측은 실 구현 상태의 정확한 연산 보드 성능 에 대한 확신은 부족하다. 이를 위한 보충을 위해서 각 단계별 시뮬레이션을 통하여 사전 검증을 한다. 하지만, 시뮬레이션은 신호처리의 일부 알고리즘 성능에 대한 검증으로 확인한다. 여전히 실 구현과의 차이가 발생해 개발 성능의 margin을 두고 설계 한다. 이런 개발절차는 신호 처리 알고리즘의 수정이나, 연동의 변경 시 개발자에게 큰 어려움으로 남아있다.

AXIS TOOL을 활용 시에 레이더 신호처리 개발에 대한 설계 margin의 폭이 줄어 들고, 시스템 설계 변경에 대한 신호처리기의 변경이 보다

유연하다. AXIS TOOL을 활용한 개발 방법을 알아보도록 한다.

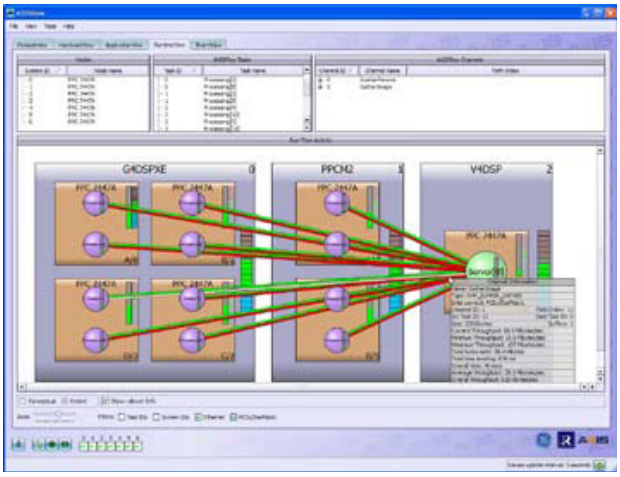
AXIS TOOL은 점진적 개발 방법론 적용할 수 있는 GUI 환경을 통하여 쉽게 적용할 수 있는 환경이다. 따라서 신호처리기의 연산 속도 및 처리 용량에 대한 분석을 기존 방식과 동일하게 고찰한 후, 레이더 신호처리의 알고리즘 흐름 구성도를 작성한다. 앞서 설명한 AXISLib를 활용한 레이더 신호처리의 최대 연산 성능을 시뮬레이션에서 검증 한 후, 그림 4와 같이 AXISView의 Application View를 통하여 실 개발 보드에 레이더 신호처리 담당하는 Task과 데이터 흐름을 관리하는 I/O buffer 및 message를 GUI를 통하여 작성한다.



(그림 4) AxisView의 Application View 예시[3]

작성된 Application View는 기본 신호처리의 흐름 제어 소스 코드가 자동적으로 생성되고, 알고리즘은 개발자가 Task별 개발하여 컴파일/링크를 통하여 신호처리기의 실행 파일이 생성된다.

이를 실행시키면, AXISView의 RunTimeView(그림 5) 화면에서 각 Task별 전송 용량과 전송시간 등을 확인한다. 이를 통하여 Task별 병목지점과 Task의 효율성에 대한 판단을 도식적으로 할 수 있다.



(그림 5) AxisView의 RuntimeView 예시[3]

다른 분석 Tool인 AXISVeiw의 EventView를 통하여 설계된 Task 단위 또는 내부의 함수 단위에서의 처리 시간을 실시간 적으로 측정할 수 있다. 이를 통하여 함수의 처리시간의 적정성 및 알고리즘의 효율성을 확인 할 수 있다.

위의 과정에서 연산 속도 및 병목 현상의 문제점을 확인하고 이를 해결하기 위해서 설계된 Task 및 IO를 연산보드 혹은 Multi-processor에 대한 재 배치를 GUI 환경을 통한 빠른 재 설계를 통하여 병목현상을 해결할 수 있다. 또한 점진적으로 설계 및 성능 확인을 반복 수행함으로써, 각 보드의 적정 연산 속도와 최적화를 향상시킬 수 있었다.

또한 AXIS TOOL을 통하여 레이더 신호처리기의 알고리즘이 변경되거나, 추가 및 삭제 되는 기능의 재배치를 쉽게 적용하고 이에 대한 소스코드를 편리하게 생성 및 실행 파일을 만들 수 있다.

3. 결론

레이더 신호처리기의 개발은 시스템의 요구사항 변경에 의해서 성능 및 설계 변경이 일어나고, 개발 중에 신호처리 알고리즘의 향상에 의해서 설계 변경이 빈번하다. 이런 구조에서 능동적으로 변경사항에 대처하기 위해선 점진적 개발 접근의 필요성을 요구된다. 이를 지원하는 개발 AXIS TOOL을 적극적으로 사용함으로 개발 성능의 효율성과 개발의 편리성을 도모할 수 있다. 물론 TOOL에 대한 학습은 무한한 노력은 필요하다. 본 논문에선 레이더 신호처리, 즉 신호의 흐름이 파이프라인과 같은 Frame 구조에서 적용하였다. 하지만 다양한 과제에서 각각의 개발 Frame을 갖고 있어 이 논문에서 소개된 개발 성격과 다를 수 있다. 일괄적으로 적용 할 수 있는 없기 때문에 많은 부분에서 과제에 적합한 검증이 선행 되어야 할 것 같다.

참고문헌

[1] Thomas W. Jeffrey. “Phased-Array Radar Design" Scitech
[2] GE intelligent Platform "AXIS Overview”
[3] GE intelligent Platform "AXIS -Taking the sting out of Multiprocessor DSP Application"