工學碩士 學位論文

핸즈프리 통신을 위한 다중 채널 음성 픽업 임베디드 시스템 설계

A Design of Multi-channel Speech Pickup Embedded System for Hands-free Communication

指導敎授 金 基 萬

2006年 2月

韓國海洋大學校 大學院

電 波 工 學 科

朱 炯 俊

차 례

그 림 차 례

기 호 표

$d(t)$	원하는 신호
$e(t)$	원하는 신호와 출력 신호 사이의 차이
$\mathbf{i}(t)$	간섭 신호 벡터
I	Zero-Padding 인수
$\mathbf{n}(t)$	순수 잡음 벡터
$\mathbf{R}_{uu}$	원하지 않는 모든 신호 성분의 상관 행렬
w	배열 시스템의 특성 가중치
$x_i(t)$	i 번째 마이크로폰에서 수신되는 신호
$y_i[n]$	i 번째 마이크로폰의 입력 신호
σ_s^2	신호의 파워
τ	마이크로폰 사이의 시간 지연
ω_0	캐리어 주파수

약 어 표

CORDIC	Coordinate Rotation Digital Computer
FPGA	Field Programmable Gate Array
LE	Logic Elements
LUT	Look-Up Table
MMSE	Minimum Mean Square Error
MV	Minimum Variance
PE	Processing Elements
SINR	Signal-to-Interference plus Noise Ratio

ABSTRACT

Recently, The hands-free communication systems are required for the safety of driving in the environment of inside noisy automobile. Among the most popular hands-free algorithm, array processing algorithm is most widely used. Since the primary advantage of using an array is to enhance a desired signal and reject jamming interferences, array signal processing is essential to satisfy demand of user. In general, array beamforming algorithm is a spatial filtering operation performed on the data received by an array of sensors, such as antennas, hydrophones, or microphones. It provides a system with the ability to “listen” directionally even when the individual microphone in the array are omnidirectional. Therefore, in this thesis, the multi-channel speech pickup system using the array beamforming algorithm for enhancement of calling quality is presented. An FPGA system has better performance than any other system, which is multiprocessing systems with high-performance DSPs. This advantage is due to the simplicity of the core calculation, the limitations of the DSP’s address calculation hardware, and the ability to customize the I/O of the FPGA to the application. For real-time operation, the enhanced speech pickup (beamforming) hardware must calculate all of the beams of interest for each set of new samples; in other words, as implied by the computational requirements discussed earlier, all

beams must be calculated at the sensor sample rate.

Therefore this thesis implements speech pickup system using the Nios-II processor with real-time I/O data processing speed. A Nios-II processor system is equivalent to a microcontroller or “computer on a chip” that includes a CPU and a combination of peripherals and memory on a single chip. Furthermore, this thesis describe method of designed using on-chip peripherals, and interfaces to off-chip memories and peripherals (SRAM, Flash memory, DMA, et al.).

To verify the effectiveness of implemented speech pickup system on Nios-II processor, the results of Niso-II processor are compare with results of computer simulation (MATLAB) and conventional DSP processor (TMS320C6711). According to the results of the speech pickup system on Nios-II processor showed a good agreement with those of computer simulation (MATLAB) and conventional DSP processor (TMS320C6711).

제 1 장 서 론

지난 몇 년간 모바일폰은 일반 대중들에게 널리 보급되어왔다. 이미 대중화가 되어버린 이러한 모바일폰은 그 사용에 있어서 물리적, 정신적으로 많은 문제점들을 가져오게 되었다. 특히 차량 내에서 모바일폰을 사용할 경우 핸드폰의 안테나로부터 발생하는 전자파에 따른 피해와 운전에 대한 집중력 감소는 교통사고를 야기할 수 있는 큰 문제이다. 그에 따라 차량 내에서 이를 해결하기 위한 방법이 핸즈프리 통신(Hands Free Communication)이지만 현재의 핸즈프리 통신은 주변잡음의 영향을 크게 받아 통신에 많은 장애가 따르는 문제점을 가지고 있다[1].

따라서 이러한 통신 장애를 극복하기 위해서는 주변잡음을 제거하고 원하는 음성만을 픽업하여 통신을 하는 기법이 필요하다. 그래서 다수의 마이크로폰 어레이를 이용하여 이러한 문제점을 극복할 수 있는 방안이 모색되었다. 센서 어레이는 수십년간 많은 신호처리 응용 분야에서 사용되어왔다. 레이더와 소나 시스템에서 어레이는 목표물의 검출뿐만 아니라 위치추적을 위해 사용되었다. 그리고 지진파 어레이는 석유탐사를 위해 사용되었으며, 안테나 어레이는 텔레통신 채널의 용량을 늘리기 위해 사용되었다[2].

주변잡음을 제거하고 음성 통화품질을 향상시키기 위해서 마이크로폰 어레이를 사용하는 빔 형성기를 이용한다. 일반적으로 빔 형성기에는 관찰하는 음장안에서 2 개 이상의 신호와 잡음으로 인한 영향을 줄이기 위해 공간적으로 선형 필터링을 추가하여 각 센서의 출력을 여과하는 Filter-and-Sum 빔 형성기와 입력 신호를 주파수 영역에서 계산 수행하는 주파수 영역 빔 형성기가 있으며, 신호가

마이크로폰에 도달하는 전파시간으로 신호원을 간단히 구하는 Delay-and-Sum 빔 형성기가 있다[3]. 또한 이러한 기본적인 빔 형성 기법 이외에도 좀 더 성능을 향상시키기 위한 적응 빔 형성 기법들에 대한 연구도 진행되고 있다. 대표적인 알고리즘들로는 MMSE(Mean square error), MV(Minimum variance), SINR(Signal to interference plus noise ratio), ML(Maximum likelihood) 기법 등이 있다[2]. 본 논문에서는 음성 신호 취득을 위하여 가장 대표적이고 하드웨어로 구현했을 때 그 알고리즘 접근이 용이한 Delay-and-Sum 빔 형성기를 설계하였다. 특히 초기에는 DSP 프로세서 만을 이용한 설계가 이루어져 왔으나, 최근 FPGA 제작 기술의 발전에 따라 FPGA 를 이용한 빔 형성기 설계가 일부 이루어지고 있다. 이는 하드웨어 집적기술의 발달로 응용 알고리즘을 하드웨어로 구성하는데 있어서 빔 형성기의 경우처럼 제어를 많이 받지 않는 독립적인 계산 알고리즘의 경우는 고사양의 FPGA 로 구현이 가능하기 때문이다. 특히 최근 응용 프로그램에 활용할 수 있는 Gate Array 의 집적도가 높아지고 가격이 저렴해지면서 그에 대한 관심도는 극대화되고 있다[4]. 1995 년 Russell J.는 기본적인 디지털 신호처리 알고리즘들을 FPGA 에 적용시킴으로써 그 성능들을 비교하였다[5]. 그로부터 1998 년 Paul Graham 은 소나 신호처리를 위해 FPGA Multicomputer Board 를 개발하여 빔 형성기를 구현하였으며[6], 이후 CORDIC 알고리즘에 기반한 FPGA 빔 형성기를 설계하여, 기존의 75% 로직 성분으로 시스템을 구현하여 DSP 프로세서를 이용한 빔 형성기와 비교하였다[7]. 국내에서는 최근에서야 그 연구 결과가 발표되고 있는데, 아날로그 디바이스 사의 SHARC 계열의 DSP 를 적용하여 실시간 처리가 가능한 빔 형성기를 구현한 사례가 있다[4].

본 논문에서는 zero-padding 보간법을 사용하여 해상도를 높임으로써 입력 받은 데이터의 음질을 개선하는 빔 형성기를

설계하였다. 또한 실시간 데이터 처리와 구현된 하드웨어의 집적성을 위하여 FPGA 로써 시스템을 구현하였다. 그러나 단순한 FPGA 만으로는 빔 형성기의 실시간 처리가 어려우며 데이터 처리 과정에서의 해상도가 떨어지게 된다. 따라서 본 논문에서 구현한 디바이스는 ALTERA 사의 Nios-II Processor 이며, 이는 Gate Array 뿐만 아니라 DSP 블록과 자체 프로세서 코어를 제공한다. 이러한 Nios-II Processor 의 이용을 통해 데이터 입출력의 효율성 상승뿐 아니라, 다른 프로세서 등을 이용해 구현한 시스템보다 성능 대 비용의 효율을 증가시킬 수 있는 장점이 있으며, 실제 프로세서 설계 시간 또한 단축시킬 수 있다.

본 논문의 구성은 제 1 장 서론에 이어, 제 2 장에서는 여러 가지 빔 형성 기법과 본 논문에서 사용된 빔 형성기에 대한 설명을 하고, 제 3 장에서는 기존의 빔 형성기 하드웨어 제작 기법에 대해 소개한다. 제 4 장에서는 FPGA 구현 과정과 그 결과를 언급하였고, 마지막으로 제 5 장에서 결론 및 향후 연구 방향을 제시함으로써 본 논문의 끝을 맺는다.

제 2 장 빔 형성 알고리즘

어레이 신호처리의 핵심은 원하는 신호를 개선하고 간섭과 잡음을 줄이는 것이다. 그에 따라 최대의 배열 이득을 얻기 위해 다양한 기법들이 사용되어 왔다. 그래서 본 장에서는 기존에 연구되었던 빔 형성 기법들과 적응 빔 형성 기법들에 대하여 간단히 소개한다.

· Delay-and-Sum(Time-domain) Beamforming
· Frequency-domain Beamforming
· Minimum Mean square error(MMSE) criterion
· Minimum variance(MV) criterion
· Signal to interference plus noise ratio(SINR) criterion

제 2-1 절 등 간격의 선형 배열에서의 빔 형성 기법

본 절에서는 등 간격으로 배치된 선형 배열에서의 기본 빔 형성 기법인 Delay-and-Sum 빔 형성 기법과 주파수 영역 빔 형성 기법에 대해 간단히 설명한다.

2-1-1. Delay-and-Sum 빔 형성 기법

Delay-and-Sum 빔 형성 기법은 어레이 신호처리 알고리즘 가운데 가장 오래되고 간단한 기본 알고리즘 구조이다. 음원으로부터

방사된 신호는 마이크로폰 배열까지 전파되는데 있어서 각각의 상대적인 시간지연을 가지고 수신된다. 그림 2-1 은 음원에서 발생한 신호가 마이크로폰과의 거리 차이만큼 시간 지연된 상태로 입력되는 것을 보여준다.

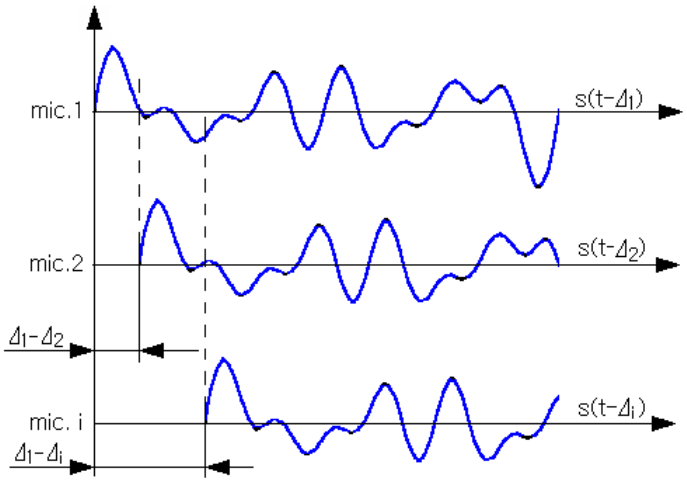


그림 2-1. 마이크로폰에 수신된 신호의 시간지연

Fig. 2-1. Time Delay of Received Signal at Microphone.

이와 같은 마이크로폰의 입력 값은 식(2.1)과 같이 나타낼 수 있다.

$$y_i[n] = y[n - D_i] + v_i[n] \qquad (2.1)$$

여기서 i는 마이크로폰 번호, D_i는 시간지연, v_i는 잡음을 의미한다. 이렇게 시간 지연을 가지고 입력된 신호들은 해당하는 지연 동작을 거친 후 가중치를 곱하고, 모든 채널에 대한 신호를 더함으로써 빔 출력을 얻을 수 있다. 그 관계식은 식 (2.2)와 같다.

$$\tilde{y}_i[n] = \frac{1}{N}\sum_{i=0}^{N-1} y_i[n+D_i] = y[n]+v[n] \qquad (2.2)$$

여기서 $v[n]$은 부가 잡음이다. 이러한 Delay-and-Sum 빔 형성기의 구조는 그림 2-2와 같다.

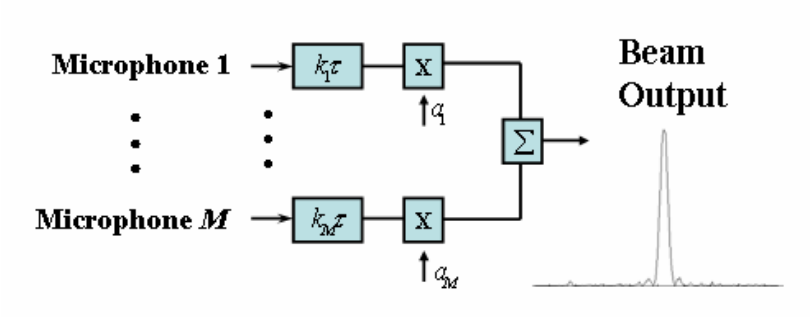


그림 2-2. Delay-and-Sum 빔 형성기의 블록 다이어그램

Fig. 2-2. Block diagram of Delay-and-Sum Beamformer.

2-1-2. 주파수 영역 빔 형성 기법

주파수 영역 빔 형성 기법은 간단히 마이크로폰으로 입력된 신호를 주파수 영역 안에서 계산하는 것이다. 그림 2-3에 N 개의 마이크로폰 배열에서의 입사 신호에 대한 표현이 나타나 있다.

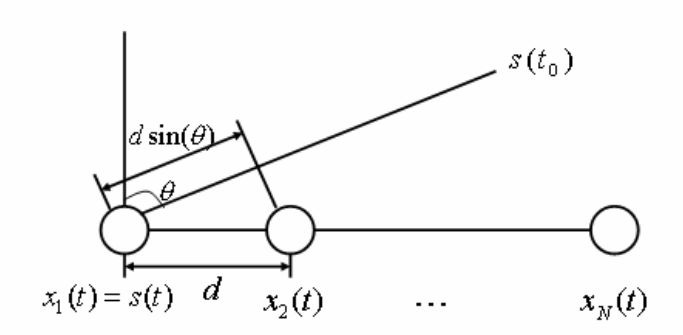


그림 2-3. 등 간격 선형 배열에서의 신호 모델

Fig. 2-3. Singnal Model for Uniform Linear Array.

기준 채널인 1번 마이크로폰에서 수신된 신호는 식(2.3)과 같이 표현된다.

$$x_1(t) = \sigma_s \cdot e^{j\omega_0 t} = s(t) \tag{2.3}$$

여기서 ω_0는 캐리어 주파수이며 σ_s^2은 신호의 파워를 말한다. 두 번째 마이크로폰에서의 출력 $x_2(t)$는 식 (2.4)의 시간 지연에 의해 표현된다.

$$\tau = \frac{d\sin(\theta)}{c} \tag{2.4}$$

여기서 c는 전파 속도이다. 따라서 $x_2(t)$는 식 (2.5)와 같다.

$$\begin{aligned} x_2(t) &= x_1(t+\tau) \\ &= x_1(t)\cdot e^{j\omega_0 \frac{d\sin(\theta)}{c}} \\ &= s(t)\cdot e^{j2\pi \frac{d\sin(\theta)}{\lambda}} \end{aligned} \tag{2.5}$$

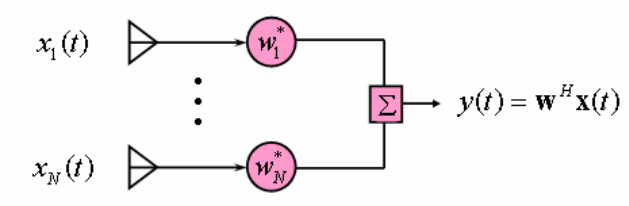


그림 2-4. 일반적인 협대역 배열 시스템

Fig. 2-4. A General Narrowband Array System.

여기서 λ 는 반송(carrier)신호의 파장이다. 식 (2.5)를 이용하여 i 번째 마이크로폰은 식 (2.6)과 같이 표현된다.

$$x_i(t) = s(t) \cdot e^{j2\pi \frac{(i-1)d\sin(\theta)}{\lambda}} + n_i(t) \tag{2.6}$$

일반적인 배열 시스템은 가중치를 곱하고 더해줌으로써 구현된다. 즉, 그림 2-4 에서 보는 바와 같이 공간 필터링 동작을 수행하는 것이다. 배열의 출력은 다음과 같이 표현된다.

$$y(t) = \mathbf{w}^H \mathbf{x}(t) = \sum_{n=1}^{N} w_n^* x_n(t) \tag{2.7}$$

여기서, w_n^* 는 배열 시스템의 각 신호에 대한 실제 가중치를 의미하며, $\mathbf{w}^H$ 는 가중치들의 합에 대한 Hermitian matrix이다.

제 2-2 절 최적화 빔 형성 기법

본 절에서는 제 2 장의 서두에서 언급된 MMSE, MV, SINR 기법에 기반한 빔 형성 기법에 대해 간단히 설명한다. 여기서 MMSE와 MV 기법은 수신된 신호의 상관 행렬을 이용하지만, SINR 기법은 잡음의 상관 행렬을 이용하여 연산한다.

2-2-1. Minimum Mean Square Error 빔 형성기

MMSE 빔 형성기는 Wiener solution 에 기반하여 Windrow 등에 의해 제안되었다[9]. 이는 시간 영역 적응 필터에 광범위하게 사용된다. 오차는 그림 2-5 에서와 같이 원하는 신호 $d(t)$와 실제 배열의 출력 신호 $y(t)$ 사이의 차이에 의해 정의된다.

$$e(t) = d(t) - y(t) = d(t) - \mathbf{w}^H \mathbf{x}(t) \tag{2.8}$$

그리고, 최소 자승 오차 ε는 식 (2.9)와 같다.

$$\begin{aligned} \varepsilon &= E\{e^2(t)\} = E\left\{\left[d(t) - \mathbf{w}^H \mathbf{x}(t)\right]^2\right\} \\ &= E\{d^2(t)\} - 2\,\mathrm{Re}\{\mathbf{w}^H \mathbf{r}_{xd}\} + \mathbf{w}^H \mathbf{R}_{xx} \mathbf{w} \end{aligned} \tag{2.9}$$

여기서, r_{xd}의 값은 식 (2.10)과 같다.

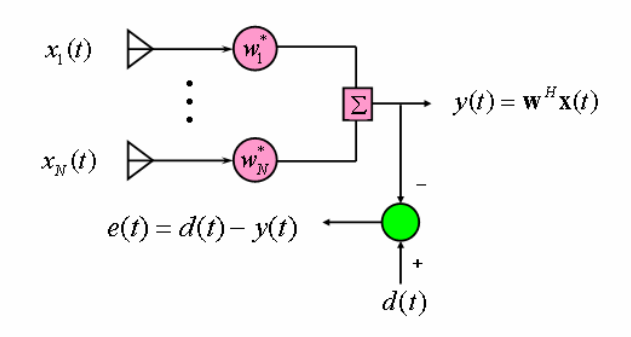


그림 2-5. MMSE(Minimum mean square error) 빔 형성기

Fig. 2-5. MMSE(Minimum mean square error) Beamformer.

$$r_{xd} = E\{x(t)d^*(t)\} = \begin{bmatrix} E\{x_1(t)d^*(t)\} \\ E\{x_2(t)d^*(t)\} \\ \vdots \\ E\{x_N(t)d^*(t)\} \end{bmatrix}, \quad \mathbf{R}_{xx} = E\{\mathbf{x}\mathbf{x}^H\} \qquad (2.10)$$

최소 자승 오차의 가중치 벡터는 $\mathbf{w}$에 대해 ε을 편미분 하여 0으로 놓음으로써 식 (2.11)과 같이 구할 수 있다.

$$0 = \frac{\partial \varepsilon}{\partial \mathbf{w}} = 2(\mathbf{R}_{xx}\mathbf{w} - \mathbf{r}_{xd}) \qquad (2.11)$$

결국, 가중치 벡터의 최적 해는 식 (2.12)와 같이 표현된다.

$$\mathbf{w} = \mathbf{R}_{xx}^{-1}\mathbf{r}_{xd} \qquad (2.12)$$

$\mathbf{r}_{xd}$는 $d(t) = s(t)$으로 하면 식 (2.13)과 같이 쓸 수 있다.

$$\mathbf{r}_{xd} = E\{\mathbf{x}(t)d^*(t)\} = \sigma_s^2 \cdot \mathbf{a}(\theta) \tag{2.13}$$

이제 식 (2.12)에서 식 (2.13)을 빼면 MMSE 빔 형성기의 최적 가중치 벡터는 식 (2.14)와 같이 표현된다.

$$\mathbf{w}_{MMSE} = \sigma_s^2 \cdot \mathbf{R}_{xx}^{-1}\mathbf{a}(\theta) \tag{2.14}$$

2-2-2. Minimum Variance 빔 형성기

MV 빔 형성기는 지향각에 대한 이득을 유지하는 동안 원하지 않는 간섭과 잡음을 줄여 배열 출력의 파워를 최소화 하는 것이다[12]. 배열 출력의 파워는 다음과 같이 표현된다.

$$E\left\{\left|y(t)\right|^2\right\} = \mathbf{w}^H E\left\{\mathbf{x}(t)\mathbf{x}^H(t)\right\}\mathbf{w} = \mathbf{w}^H\mathbf{R}_{xx}\mathbf{w} \tag{2.15}$$

또한, MV의 최적 가중치는 식 (2.16)을 풀어서 알 수 있다.

$$\min_{\mathbf{w}} \mathbf{w}^H\mathbf{R}_{xx}\mathbf{w} \qquad \textit{subject to} \qquad \mathbf{w}^H\mathbf{a}(\theta) = 1 \tag{2.16}$$

이 최소화 문제는 Lagrange multiplier를 사용하여 쉽게 풀 수 있다.
이후의 최적 가중치 벡터 $\mathbf{w}$는 식 (2.17)과 같이 주어진다.

$$\mathbf{w}_{MV} = \frac{\mathbf{R}_{xx}^{-1}\mathbf{a}(\theta)}{\mathbf{a}^{H}(\theta)\mathbf{R}_{xx}^{-1}\mathbf{a}(\theta)} \tag{2.17}$$

그리고 식 (2.15)에서 식 (2.17)을 빼면, 식 (2.18)과 같이 배열 출력 파워를 구할 수 있다.

$$P(\theta) = \frac{1}{\mathbf{a}^{H}(\theta)\mathbf{R}_{xx}^{-1}\mathbf{a}(\theta)} \tag{2.18}$$

결과적으로, 식 (2.14)의 $\mathbf{w}_{MMSE}$는 $\mathbf{w}_{MV}$의 스칼라 곱이 된다.

2-2-3. SINR을 최대화 하는 빔 형성기

가중치 벡터 $\mathbf{w}$는 빔 형성기의 출력에서 SINR을 최대화 함으로써 최적화될 수 있다. 우선, 식 (2.5)는 다음과 같이 쓸 수 있다.

$$\mathbf{x}(t) = s(t)\mathbf{a}(\theta) + \mathbf{u}(t) \tag{2.19}$$

여기에서 원하지 않는 신호 벡터 $\mathbf{u}(t)$는 순수 잡음 벡터 $\mathbf{n}(t)$에 간섭 신호 벡터 $\mathbf{i}(t)$가 더해진 형태이다. 따라서 배열 출력 신호의 원하는 신호 성분과 원하지 않는 신호 성분은 다음과 같이 주어진다.

$$y_s(t) = \mathbf{w}^{H}\mathbf{a}(\theta)s(t) \tag{2.20}$$

$$y_n(t) = \mathbf{w}^{H}\mathbf{u}(t) \tag{2.21}$$

그러므로, 출력 SINR은 식 (2.22)와 같이 쓸 수 있다.

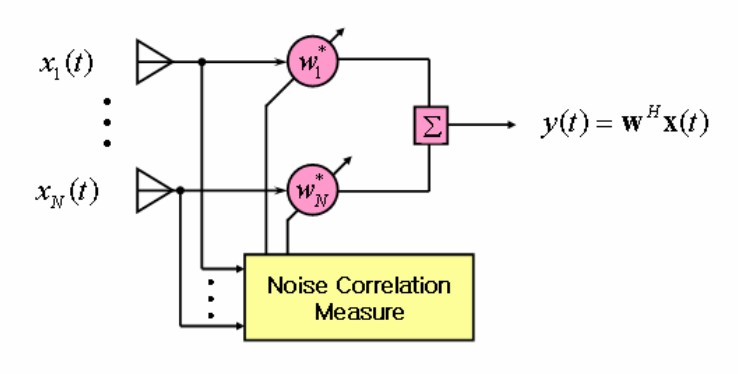


그림 2-6. SINR 빔 형성기의 블록 다이어그램

Fig. 2-6. Block diagram of SINR Beamformer.

$$SINR_{out} = \frac{E\left\{\left|\mathbf{w}^H\mathbf{a}(\theta)s(t)\right|^2\right\}}{E\left\{\left|\mathbf{w}^H\mathbf{u}(t)\right|^2\right\}} = \frac{\sigma_s^2\left|\mathbf{w}^H\mathbf{a}(\theta)\right|}{\mathbf{w}^H\mathbf{R}_{uu}\mathbf{w}} \tag{2.22}$$

여기서 $\mathbf{R}_{uu}$는 원하지 않는 모든 신호 성분의 상관 행렬이다. $SINR_{out}$의 최대화는 여러 가지 방법에 의해 풀 수 있으나, 간단하게 Cauchy-Schwarz inequality를 사용하여 구하면 식 (2.23)과 같다.

$$\begin{aligned} SINR_{out} &= \frac{\sigma_s^2\left|\left(\mathbf{R}_{uu}^{1/2}\mathbf{w}\right)^H\left(\mathbf{R}_{uu}^{-1/2}\mathbf{a}(\theta)\right)\right|^2}{\mathbf{w}^H\mathbf{R}_{uu}\mathbf{w}} \\ &\le \sigma_s^2 a^H(\theta)R_{uu}^{-1}a(\theta) \\ &= SINR_{\max} \end{aligned} \tag{2.23}$$

이와 같이 가중치 벡터도 구하면 식 (2.24)와 같다.

$$w_{SINR} = c \cdot R_{uu}^{-1} a(\theta) \qquad (2.24)$$

여기서 c는 0이 아닌 상수이다. 그림 2-6은 SINR 빔 형성기의 구조를 보여준다.

제 2-3 절 보간 필터를 갖는 빔 형성기

앞 선 절에서 기본적인 빔 형성 기법들과 적응 빔 형성 기법들의 종류를 살펴보았다. 본 절에서는 본 논문에서 설계된 보간 필터링 빔 형성기에 대해 설명한다. 보간 필터링 빔 형성기는 기본적으로 Delay-and-Sum 빔 형성기의 구조와 비슷하다. 그러나 빔 형성기의 성능을 높이기 위해서는 다량의 입력 데이터를 확보하는 것이 중요하다. 하지만, 높은 해상도와 높은 샘플링 주파수로써 데이터를 취득할 경우 그 처리에 있어서 매우 큰 용량의 메모리와 뛰어난 성능의 CPU 등이 필요하게 된다. 그래서 사용하는 방법이 보간법이다. 보간 필터를 갖는 빔 형성기는 수신된 신호의 입력 단에서 신호를 과 샘플링 하여 높은 해상도의 신호를 확보하여 준다. 즉, Delay-and-Sum 빔 형성기의 입력단에 보간 회로를 부착하여 주어 다량의 입력 데이터를 확보함으로써 출력 빔의 성능 향상을 가져오는 것이다.

2-3-1. Zero-Padding 보간법

앞에서 언급하였듯이 음질 개선 성능을 높이기 위하여 입력되는

음성 신호를 그대로 처리하기 보다는 보간법을 사용하여 해상도를 높여야 한다. DSP 처리 과정 중의 하나인 보간법은 주어진 점들을 통과하는 함수를 구하고 이웃한 위치에서의 값을 찾아내어 계산하는 과정이다. 그 종류로는 Polynomial 보간법, Newton 보간법, Spline 보간법 등이 있다. 본 논문에서는 zero-padding 보간 기법을 이용하여 빔 형성기에 적용하였다. Zero-padding 기법은 각각의 샘플 데이터 사이에 원하는 수만큼의 '0'의 값을 배치시킨다. 그리고 그것의 출력은 다시 저역통과 필터를 통해 원래의 샘플 사이의 값을 보정하게 된다. 시간영역에서 샘플들 사이에 샘플들 사이에 '0'의 값으로 분리되는 간격은 식 (2.25)와 같다.

$$T_s / I = 1/(I \times f_s) \tag{2.25}$$

여기서 I는 Zero-padding의 인수 성분이다. 결국, 보간법에 의해 과샘플링 되어 그 샘플율은 $f_s \times I$가 된다. Zero-padding 보간법의 개념은 그림 2-7에 잘 나와 있다. 시간 영역에서 샘플 주기 $T_s = 500$인 원래 신호가 (a)에 나와 있으며, Zero-padding의 인수 $I = 4$에 의해 샘플 주기 $T_s = 125$이 되도록 샘플 사이에 '0' 데이터가 깔린다. 과 샘플된 이 신호는 저역 통과 필터를 통과 하면서 그림 2-7의 (c)와 같이 보간된다.

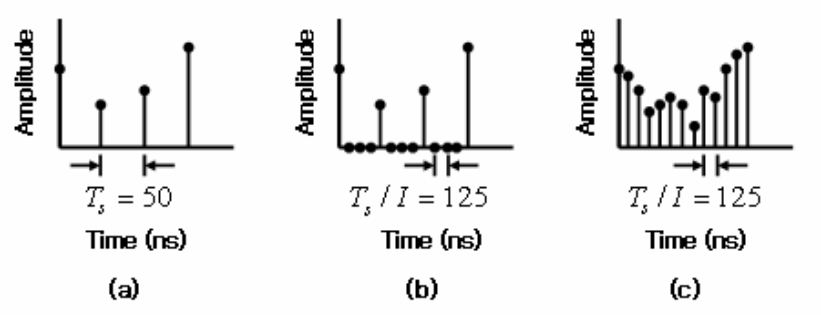


그림 2-7. Zero-padding 보간법의 시간영역에서의 표현 ($I = 4$),
(a) 원 신호, (b) Zero-Padding된 신호, (c) 필터링된 신호

Fig. 2-7. Time domain Representations of Zero-padding Interpolation ($I = 4$), (a) Original Signal, (b) Zero-Padded Signal, (c) Filtered Signal.

2-3-2. 보간 필터를 갖는 Delay-and-Sum 빔 형성기

빔 형성기는 M개의 마이크로폰 배열로부터 공간상의 신호를 수신하여 빔 형성 알고리즘을 거쳐 출력을 얻게 된다. 본 논문에서는 차량 내 핸즈프리 통신을 위한 빔 형성기를 설계하였다. 그에 따라 차량 내에서 음성 신호를 수신하여 연산을 하게 된다. 일반적으로 음성의 경우 코덱에서 16비트 양자화와 8kHz 샘플링을 거치게 된다. 이러한 입력 신호를 양자화 비트와 샘플링 주파수 측면에서 해상도를 향상 시키도록 구성된 것이 보간 필터링 Delay-and-Sum 빔 형성기이다. 이는 앞서 설명된 zero-padding 보간법과 2-1-1 절의 Delay-and-Sum 빔 형성기를 연결함으로써 설계할 수 있다. 보간 필터링 Delay-and-Sum 빔 형성기의 구조는 그림 2-8과 같다.

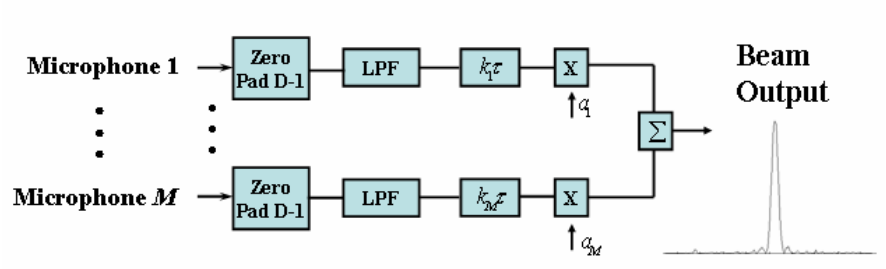


그림 2-8. 보간 필터링 된 Delay-and-Sum 빔 형성기의 블록 다이어그램

Fig. 2-8. Block diagram of Interpolation Filtered Delay-and-Sum Beamformer.

제 3 장 빔 형성기의 하드웨어 설계 기법

제 3장에서는 제 2 장에서 논의된 보간 필터링을 거친 Delay-and-Sum 빔 형성기의 실제 하드웨어 설계에 앞서 기존의 빔 형성기의 하드웨어 설계 방법과 구조에 대해 알아보았다.

제 3-1 절 SHARC DSP로 구현한 빔 형성기[6]

빔 형성기의 구조를 살펴보면 시간 지연의 처리 과정이나 가중치를 더해주는 과정에서 실제로 곱셈 연산이 많이 들어가 있음을 알 수 있다. DSP 프로세서는 곱셈 연산 수행 동작에 최적화 되어 있다. 이러한 장점을 가지는 DSP군을 이용한 기존의 빔 형성기 구조를 살펴보면 그림 3-1과 같다.

센서로부터 입력되는 다중채널의 데이터는 PCI Bus를 통해 하나

의 DSP로 전달되고 이 데이터는 DSP간의 연결인 Link port를 통해 다중의 DSP군으로 분산된다. 전달된 데이터들은 부분합 빔 동작을 수행한 뒤 하나의 대역에 대한 통합 빔을 형성하기 위하여 다시 하나의 DSP로 모이게 된다. 이 과정 역시 Link port를 통해서 이동된다. 통합된 빔은 탐지 및 식별을 위하여 또 다른 DSP군으로 넘겨지는데 이 때는 시스템 상에 묶여진 Network Link port를 이용한다. Network Link 란 신호처리 시스템의 모든 DSP가 공유하여 사용하는 Link 연결을 의미한다. 빔 형성기와 탐지 식별기 사이의 Network Link 연결로 인하여 신호처리 시스템이 정상적으로 동작하기 위해서는 코어 프로세싱과 데이터 전송프로그램들과의 시간, 할당량 등의 조율이 필요하다.

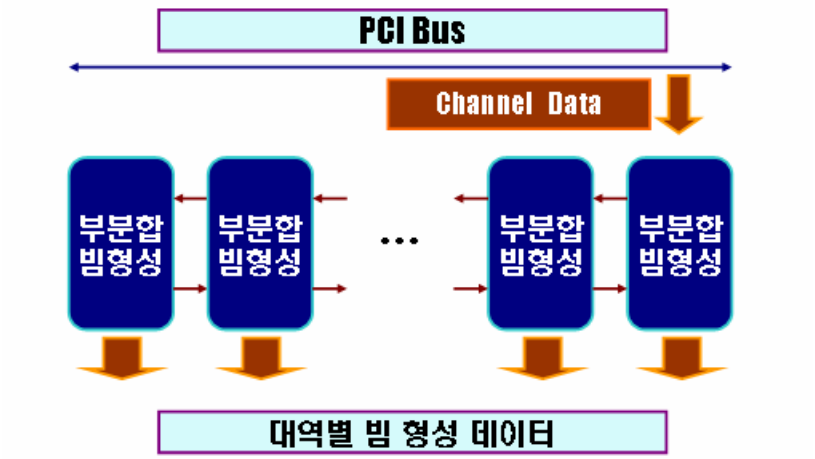


그림 3-1. DSP군을 이용한 시간영역 빔 형성기 구조

Fig. 3-1. An Architecture of Time-domain Beamformer using DSPs.

SHARC DSP의 경우 단일 명령에서 부동 소수점 혹은 사이클당 3번의 32비트 고정 소수점 곱셈과 덧셈이 가능하여 이러한 곱셈 연산을 효율적으로 할 수 있도록 최적화 되어 있다. 이렇듯 SHARC DSP는 모든 사이클에 대해서 부동 소수점 곱셈 연산을 할 수 있다.
SHAR 프로세서의 최대 연산 용량을 위해 DSP는 효과적인 메모리 시스템을 가져야만 한다. 그에 따라 각각의 SHARC는 동등한 크기의 듀얼 포트 온 침 SRAM을 가지고 있다. 이는 사이클 당 2회의 연산으로 빠르게 데이터를 접속할 수 있게 해준다.

이러한 SHARC DSP로 Delay-and-Sum 빔 형성기를 설계할 경우 몇 가지 과제를 안게 된다. 단일 사이클에서 SHARC DSP는 부동 소수점 곱셈과 덧셈을 수행할 수 있지만, 이러한 곱셈 연산을 원활이 수행하기 위해선 충분한 데이터를 확보할 수 있어야 한다. 그러나 불행히도 어드레스 재생 로직의 한계로 인하여 Delay-and-Sum 빔 형성 MAC은 모든 사이클에서 수행될 수 없다. 최소치를 따져봤을 때 이는 세 번의 사이클까지 가능하다. 그에 따라 더 많은 외부 메모리와 'Pipeline' 형태로의 DSP간 연결이 필요하게 된다.

제 3-2 절 FPGA 멀티 컴퓨터 보드를 이용한 빔 형성기[6]

앞의 절에서 살펴본 SHARC DSP를 이용한 빔 형성기를 보완하기 위해 FPGA를 이용한 빔 형성기의 설계에 대한 연구가 이루어지고 있다. 그 첫 번째로 소개할 빔 형성기는 FPGA Multicomputer Board를 이용한 빔 형성기이다. 이 빔 형성기는 크게 네 개의 주요 부분을 가진다. 먼저, PE(Processing Elements)는 현재의 빔과 센서의 사용에 센서 히스토리 메모리로 오프셋을 결정해야만 한다는 것이다. 이는 시간지연

을 연산해야 하며 일반적으로 LUT(Look-Up Table)을 사용한다. 이로부터 센서의 구조와 전파의 형태를 연산해야 한다. 일단 샘플의 위치를 알게 되면 그것은 메모리로부터 유추된다. 세 번째 단계는 shade 성분을 유추하는 것이며, 마지막으로 센서의 값이 shade 성분에 의해 곱해지고 연산되는 것이다. 이 과정은 단일 빔을 생성하는 모든 센서들에 의해 반복된다. 실시간 동작을 위해 빔 형성 하드웨어는 새로운 샘플들의 각 형태에 관심 빔의 모든 연산을 해야만 한다. 즉, 모든 빔들은 센서 샘플율에서 연산되어야 한다는 것이다. 이러한 일련의 과정은 그림 3-2와 같다.

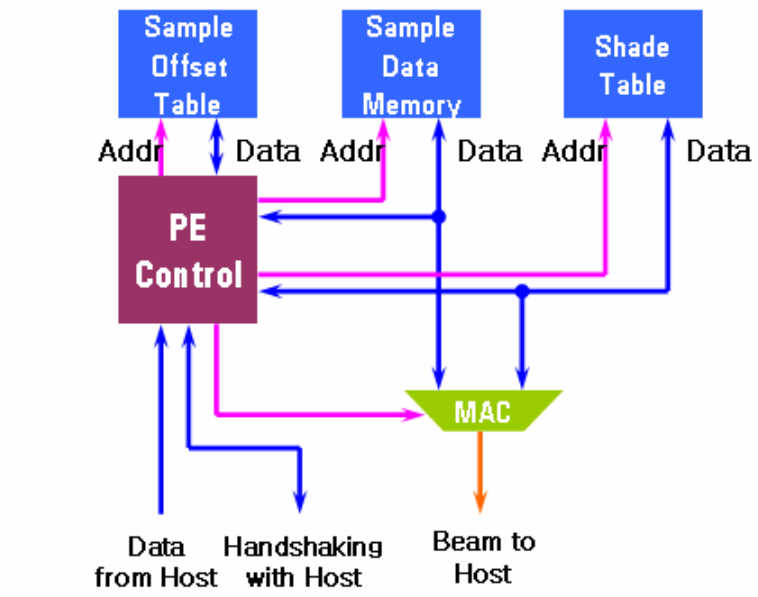


그림 3-2. 빔 형성기 프로세서 블록 다이어그램

Fig. 3-2. Beamforming Processor Block diagram.

이 빔 형성기 설계를 위한 타겟 플랫폼은 가상의 FPGA 연산 보드이다. 보드의 구조는 그림 3-3 과 같다. 각 각의 보드는 Myrinet과 같이 빠른 인터페이스와 low-latency 네트워크를 가진다. 네트워크에서의 연결은 버스 구조와는 반대로 두 점 사이에서 연결된다. 전체 보드는 하나의 Myrinet 인터페이스 프로세서와 하나의 호스트 FPGA, 네 개의 프로세싱 FPGA를 가진다. 연산과 데이터의 전송은 호스트 FPGA가 각 네 개의 FPGA들을 원 형태로 묶어서 버스로 할당을 해준다. 그 외의 버스들은 다른 디지털 장비를 가진 인터페이스나 확장 연산 보드에 직접적으로 연결할 수 있도록 해준다.

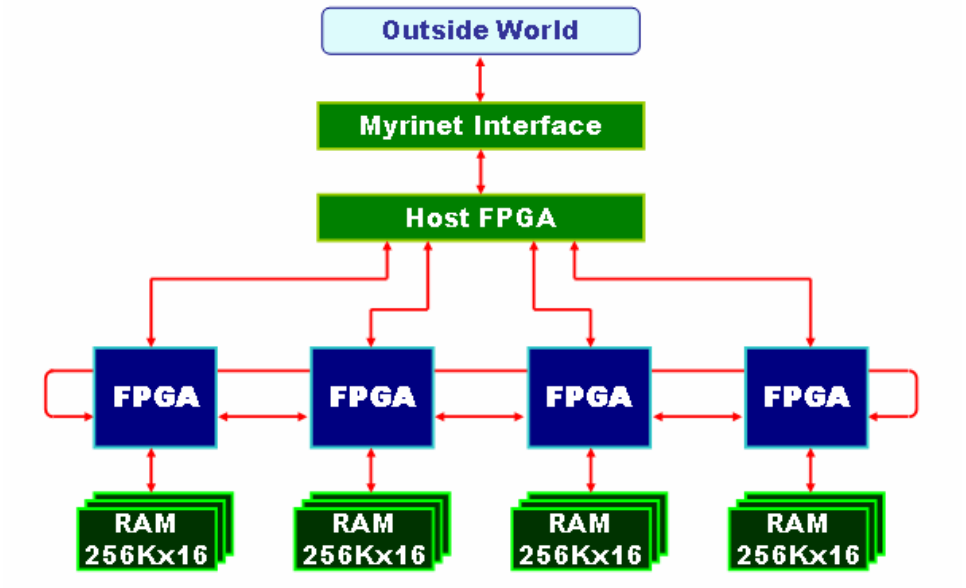


그림 3-3. FPGA 멀티 컴퓨터 보드

Fig. 3-3. FPGA Multicomputer Board.

제 3-3 절 Two-level 멀티 컴퓨터를 이용한 빔 형성기[10]

본 절에서 소개하는 Two-level 멀티 컴퓨터를 이용한 빔 형성기는 제 3-2 절에서 소개한 FPGA 멀티 컴퓨터 보드를 이용한 빔 형성기의 적용 구조이다. 제 3-1절에서 설명한대로 빔 형성기는 FPGA 매핑만으로는 내부 루프 연산 알고리즘을 수행할 수 없다. 그래서 DSP와 함께 시스템을 제작해야 하며 더욱 개선된 성능을 위해서는 멀티 프로세싱 DSP 시스템을 고려해야 한다. 그에 따라 CORDIC(Coordinate Rotation Digital Computer) 알고리즘을 사용하여 연산을 단순화 시킨 것이다. 막대한 양의 연산 문제를 해결하기 위해 선택한 모델이 Two-level

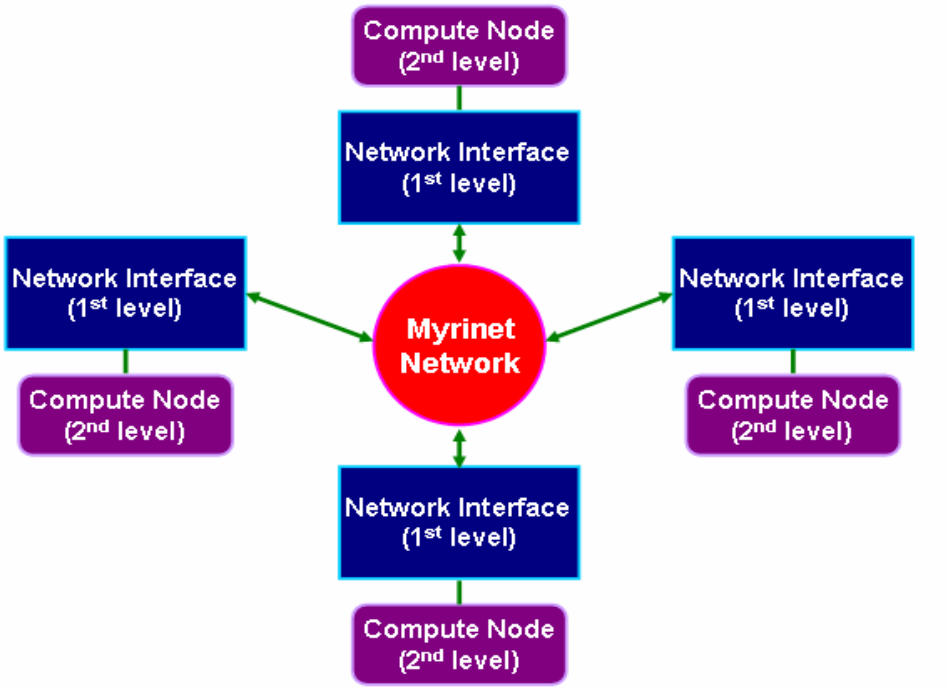


그림 3-4. Two-Level 멀티 컴퓨터

Fig. 3-4. Two-Level Multicomputer.

Multicomputer 이다. 그 구조는 그림 3-4 에 나타나 있다. 프로세서의 1st level은 Myrinet이라는 고속 네트워크를 통해 연결되었다. 이러한 프로세서는 2nd level 프로세서에 제공되는 네트워킹이나 다른 서비스들의 모든 통신을 제어한다. 즉 2nd level 프로세서는 사용자가 지정한 연산을 수행하게 된다. 이 Two-Level 멀티 컴퓨터는 결국 앞 절에서 소개한 FPGA 멀티 컴퓨터 보드에 적용되는 것이다. 제 3-2 절에서 소개된 시스템의 경우 메모리 포트에 한계를 가지고 있다. 그러나 본 절에서 소개된 시스템은 60%의 로직만으로 동일한 디자인을 구성할 수 있다. 시스템의 계산 소요량을 따져보면 다음과 같다.

· 2000Hz × 400 sensors × 10,000 beams = 8 billion MACs / second.

이 결과는 3-2 절의 시스템의 경우 최소 25개의 보드가 필요하게 된다.

제 3-4 절 DSP를 연동한 FPGA 빔 형성기[4]

DSP를 연동한 빔 형성기의 원리는 아주 간단하다. 빔 형성기는 센서로부터 다중채널의 데이터를 PCI Bus를 통해 메모리로 저장을 하게 된다. 이는 보간 처리를 한 이후 DPRAM에 저장되어 빔 안정화 과정을 거쳐 도출된 빔 연산선을 참조하여 빔 형성을 시작한다. 빔 형성기는 다중의 MAC(Multiply Accumulate) 연산자로 구성되며, 형성된 빔 데이터는 TigerSHARC Link port, 즉 DSP 단을 통해 신호처리단과 연동된다. 사용된 DSP는 8개의 DSP가 4개씩 짝지어져 2개의 Cluster로 구성된 보드들로 구현된다. FPGA로 구현되는 시간영역의 빔 형성기는 그림 3-5와 같은 개념으로 설계한다.

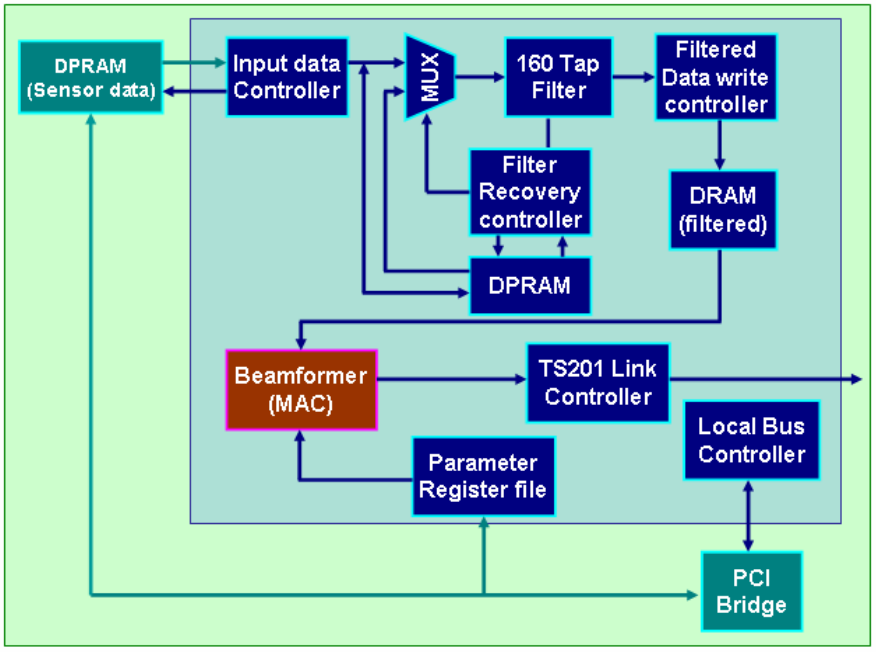


그림 3-5. 시간 영역 빔 형성기 FPGA 설계 블록도

Fig. 3-5. FPGA Design Block Diagram of Time-domain Beamformer.

제 4 장 보간 필터를 갖는 Delay-and-Sum 빔 형성기의 FPGA 설계

앞 장에서 살펴본 보간 필터를 갖는 빔 형성기를 실제 FPGA로 구현하고 그 성능을 검증하였다. 본 논문에서 설계된 빔 형성기는 4 채널의 마이크로폰으로부터 입력을 받도록 설계되었다. 핸즈프리 통신을 위한 빔 형성기이므로 일반적인 음성의 사양에 맞춰서 설계가 이루어졌다. 음성의 경우 코덱에서 16비트 양자화와 8kHz 샘플링을 거치게 된다. 하지만, 본 논문에서는 FPGA 칩의 LE(Logic Element) 비대화의 이유로 8비트 양자화된 샘플을 입력 받도록 설계하였다. 하지만, zero-padding 보간법을 통해서 신호의 양자화 비트를 16비트로 상승시키고, zero-padding 인수 $I=4$ 로 설정함으로써 샘플링 주파수를 네 배로 올렸다. 그에 따른 데이터의 흐름은 그림 4-1 과 같다.

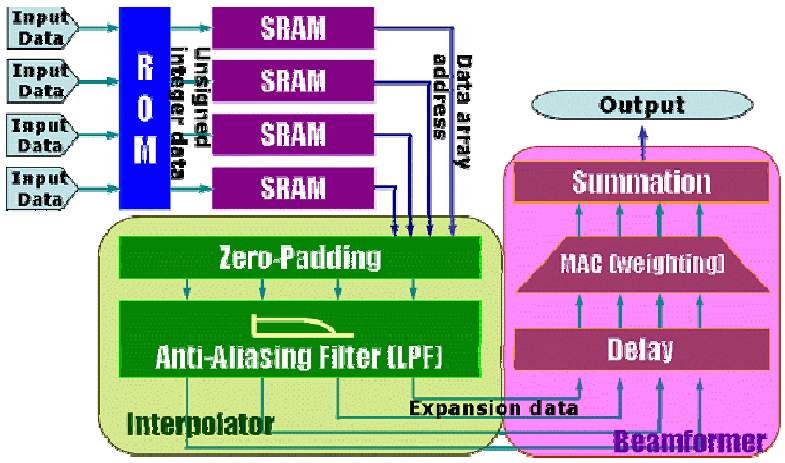


그림 4-1. 보간 필터를 갖는 Delay-and-Sum 빔 형성기의 데이터 흐름

Fig. 4-1. Data flow of Interpolated Delay-and-Sum Beamformer.

4 채널로 입력된 신호는 SRAM 에 잠시 저장되어 Zero-padding 보간법을 거쳐 데이터 해상도를 높인다. 그 신호는 다시 시간 지연 연산을 거친 후 합해져 출력 값을 얻게 되는 것이다.

제 4-1 절 빔 형성기를 위한 프로세서 설계

제 3 장에서 DSP와 FPGA를 이용한 여러 가지 빔 형성기 설계 기법들에 대해 살펴보았다. 하지만, 소개되어온 빔 형성기들은 공통적으로 데이터 입출력이나 동작 제어 부분에서 모두 DSP 프로세서를 사용하였다. 이렇게 구성하였을 경우 설계에 앞서 처음 세웠던 사양에 맞게

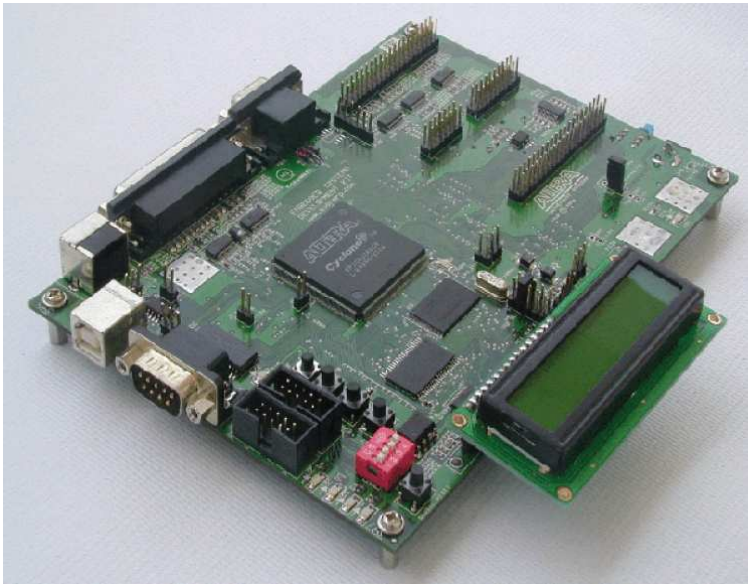


그림 4-2. Nios-II Processor UP3 Board

Fig. 4-2. Nios-II Processor UP3 Board.

어느 정도의 최적화된 하드웨어 구성을 성취할 수 있다. 그러나 사양에 맞는 프로세서를 설계하고 직접 제작하는 과정은 무척이나 까다롭고, 시간도 오래 걸리며 비용 또한 많이 들게 된다. 실제로 이러한 시스템을 위한 프로세서의 설계는 몇 주에서 몇 개월씩 걸리게 된다. 더욱이 사용 중 사양의 업그레이드나 불필요한 소자를 없애고자 할 경우 프로세서를 새로 설계하고 구입해야 하는 부담이 따르게 된다.

그에 따라 본 논문에서는 ALTERA 사의 Nios-II Embedded Processor를 사용하여 설계하였다. Nios-II Processor는 소프트 코어 프로세서로서 설계자가 원하는 바 그대로 주변장치, 메모리, 그리고 인터페이스 특성을 선택하여 사양에 맞게 프로세서를 저렴한 비용으로 커스터마이제이션 할 수 있다. 그리고 그 설계 과정도 소프트웨어를 통해 모두 제어하므로 기존의 하드 코어 프로세서의 설계 과정보다 간단하다고 할 수 있다. 또한 코어 프로세서뿐만 아니라, 높은 수행능력을 지닌

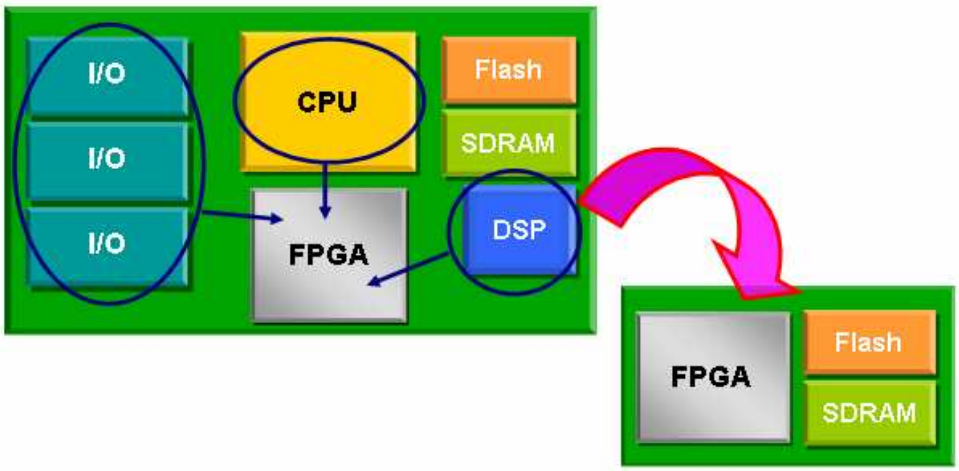


그림 4-3. Nios-II Processor의 구성 개념

Fig. 4-3. Architecture Concept of Nios-II Processor.

FPGA가 칩 내에 함께 들어있기 때문에 프로세서와 FPGA간의 연동도 같은 소프트웨어 내에서 제어할 수가 있다. Nios-II의 기본 개념을 그림 4-3에 나타내었다.

빔 형성기의 데이터를 처리하기 위해 ALTERA 사의 Nios-II Processor를 이용하여 설계하였다. 32 비트 RISC 프로세서 형태인 Nios-II의 표준 설계 블록 다이어그램은 그림 4-4와 같다. SRAM, Flash, Ethernet, MAC, LCD, LED 등의 일반적 주변장치와 ROM, Timer, UART등의 내부장치를 확인할 수 있다. 그 중 속도 향상을 위해 적용된 Nios-II의 핵심 내부 장치는 'Avalon Switch Fabric'이다. 이는 다수의 동시 데이터 전송을 가능하게 해주며 뛰어난 시스템 처리 효율을 보여준다. 전형적인 버스 구조가 그림 4-5에 있다[14].

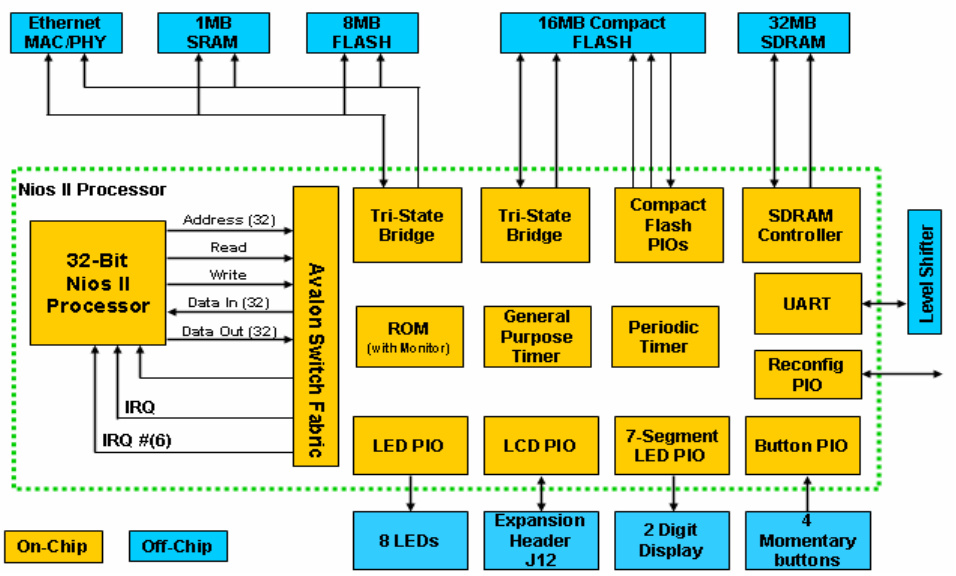


그림 4-4. Nios-II 표준 설계 블록 다이어그램

Fig. 4-4. Nios-II Standard design block diagram.

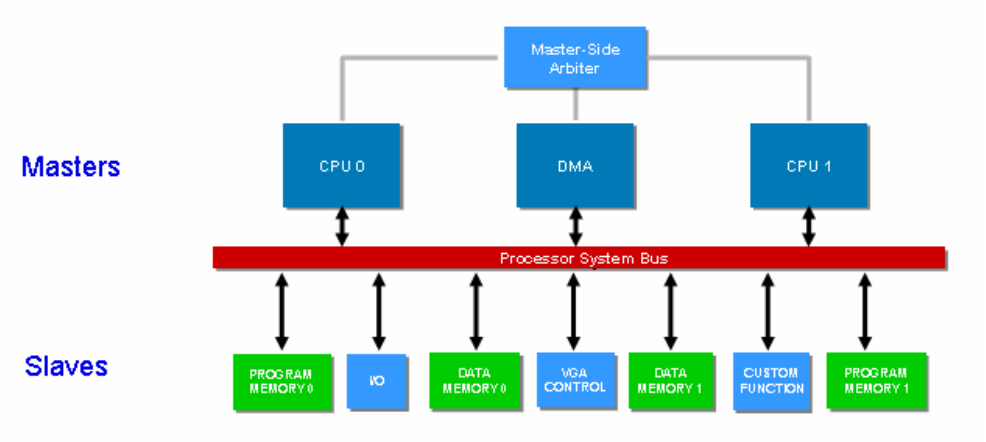


그림 4-5. 전형적인 버스 구조

Fig. 4-5. Traditional Bus Architecture.

그림에서 확인할 수 있듯이 기존의 버스 구조에서는 싱글 arbiter가 버스 마스터와 슬레이브간의 통신을 컨트롤해준다. 각각의 버스 마스터는 버스 컨트롤을 요청하면 그 후 arbiter는 버스 액세스를 싱글 마스터에게 보내준다. 만약 다수의 마스터가 한꺼번에 버스를 액세스 하려고 하면, arbiter는 고정 arbitration 규칙에 따라 하나의 마스터에게 버스 자원을 할당한다. 한 번에 하나의 마스터만이 시스템 버스와 그 자원을 액세스 할 수 있기 때문에 대역폭에 있어 병목 현상을 일으킨다. 이러한 현상은 데이터간 전송 및 연산이 실시간으로 복잡하게 진행되는 빔 형성기에 악영향을 끼치는 요소가 될 수 있다. 특히, 빔 형성기의 채널이 늘어남으로 인해 프로세서의 주변 장치들이 더욱 많이 요구될 때엔 최적화된 버스 관리가 필요로 하게 된다. 그에 따라 다음의 'Avalon Switch Fabric'는 최적화된 버스 관리가 가능하다. 'Avalon Switch Fabric'의 구조가 그림 4-6에 나타나 있다[14]. 이는 각각의 버스 마스터는 자체의 전용 내부 연결 루트를 가지기 때문에 공유 슬레이브에 대

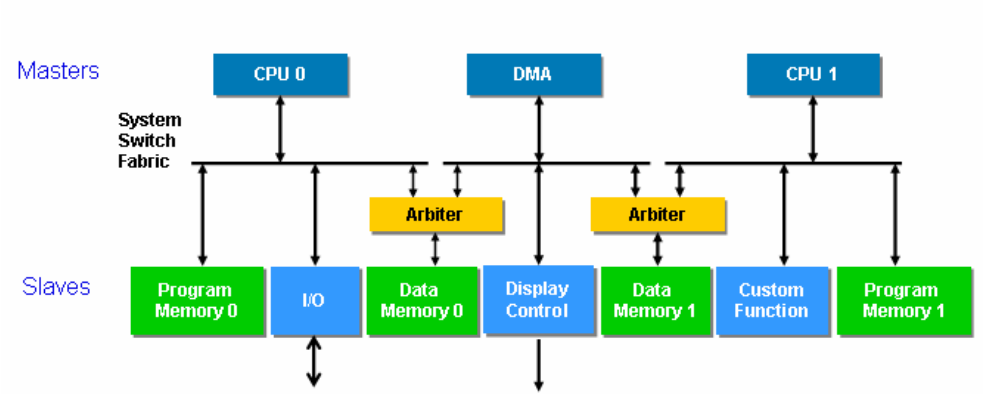


그림 4-6. Avalon Switch Fabric의 구조

Fig. 4-6. Avalon Switch Fabric Architecture.

해서만 경쟁하게 된다. 이는 부품이 추가 되거나, 주변장치의 액세스 우선순위가 바뀔 때마다 최적화된 Avalon Switch Fabric을 생성한다. 하지만, 하드웨어 리소스를 많이 소비하는 단점 역시 가지고 있다.

이러한 Nios-II에 대한 배경 하에 빔 형성기를 위한 프로세서를 QuartusII 와 SOPC Builder를 이용하여 설계하였다. 먼저 Nios II 커스텀 보드를 소프트웨어 상에서 Targeting하기 위한 보드 설계를 수행하였다. 본 논문의 설계에 사용된 보드는 ALTERA사의 협력업체에서 만든 보드로써 그 인터페이스가 Quartus II 상에 등록되어있지 않다. 그래서 설계된 시스템을 보드에 Targeting하기 위해서는 보드의 인터페이스 정보를 가지고 있는 Target 보드를 먼저 생성하여 Quartus II 에 등록을 하고 사용을 해야 한다. 그 결과를 그림 4-7에 나타내었다. 설계결과 기본 칩의 LE 점유율은 2,591/5,980으로 43%를 차지하였다. 더욱 자세한 내용은 그림 4-8에서 확인할 수 있다.

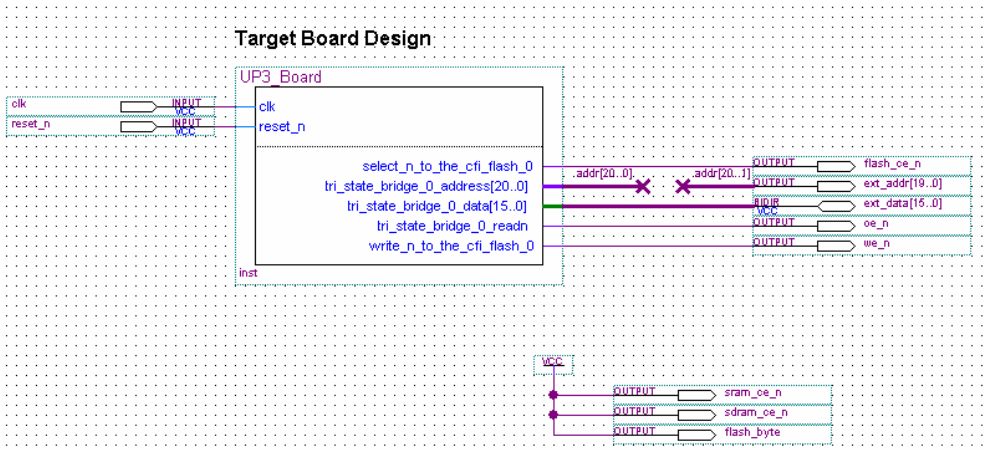


그림 4-7. UP3 Target 보드 설계

Fig. 4-7. UP3 Target board design.

Flow Status	Successful - Thu Oct 20 21:15:43 2005
Quartus II Version	5.0 Build 148 04/26/2005 SJ Full Version
Revision Name	UP3_Board
Top-level Entity Name	UP3_Board_top
Family	Cyclone
Device	EP1C6Q240C8
Timing Models	Final
Met timing requirements	Yes
Total logic elements	2,591 / 5,980 (43 %)
Total pins	44 / 185 (23 %)
Total virtual pins	0
Total memory bits	70,656 / 92,160 (76 %)
Total PLLs	0 / 2 (0 %)

그림 4-8. UP3 Target 보드의 컴파일 정보

Fig. 4-8. Flow summary of UP3 Target board.

설계에 사용된 보드는 ALTERA의 협력업체인 SLS(System Level Solution)사에서 제작한 UP3(University Program)보드로써 인터페이스를 적용하기 위한 Target 보드이므로 'Avalon Switch Fabric'과 사용자 로직과의 인터페이스를 연결해주는 'Tri-State Bridge'를 등록하였으며, 기본 리셋 동작을 담당하는 플래쉬 메모리, 그리고 SRAM과 플래쉬 메모리에 각각 Vcc를 넣어주었다. 이를 통해 커스텀 보드를 등록하고, 실제 빔 형성기에 사용될 프로세서를 설계하였다. 설계된 시스템은 입력 채널이 4개이므로 8비트 입력 단자를 4개 넣고 16비트 출력 포트를 한 개 넣었다. 그리고 더욱 효율적이고 방대한 양의 데이터를 처리하기 위한 커스텀 SRAM(IS61C6416)을 하나 더 부착하였다. 장치들간의 통신을 위해 'tri-state bridge'를 부착하고, 프로세서 내부 데이터와

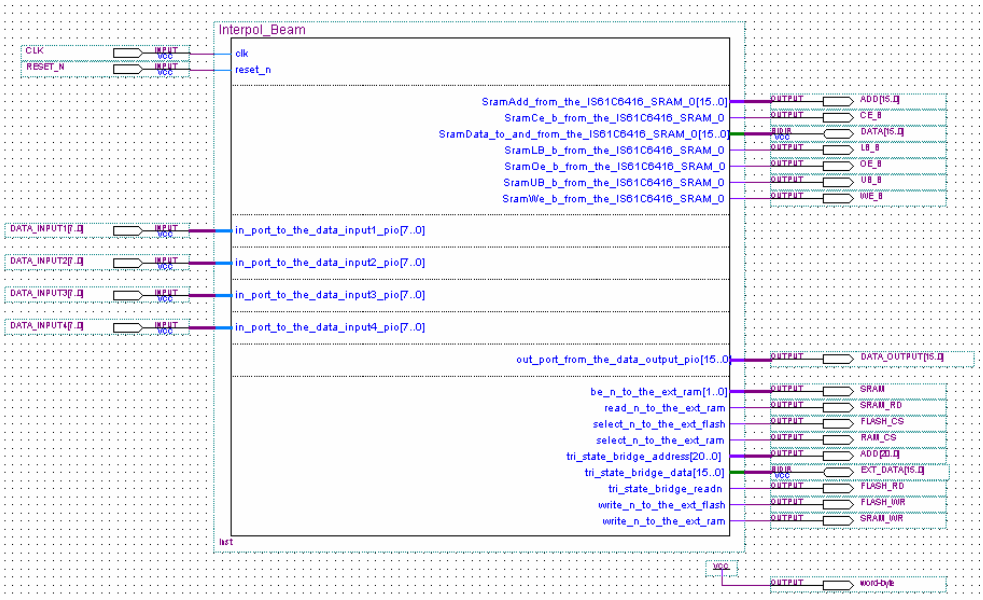


그림 4-9. 빔 형성기를 위해 개발된 Nios II Processor 시스템

Fig. 4-9. Developed Nios II Processor system for beamformer.

버스를 관리할 플래쉬와 SRAM을 부착하였다. 그리고 실질적으로 더욱 빠른 데이터 전송을 위해 DMA(Direct Memory Access) 를 부착하여 데이터 입출력 속도를 향상시켰다. 그에 따라 구성된 시스템이 그림 4-10에 있다.

설계된 시스템 중 어떤 부가장치는 UP3 보드에 적합하지 않은 장치도 있다. 그래서 본 시스템에는 어드레스를 분할하여 다시 할당해주는 장치가 필요하다. 그림 4-9에서 그 장치를 확인할 수 있다.

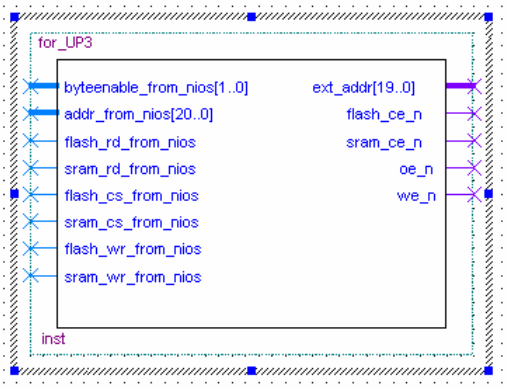


그림 4-10. UP3 보드에 최적화를 제공하는 회로

Fig. 4-10. UP3 supported circuit.

UP3 보드에 최적화를 제공하는 회로를 부착한 이후 최종 설계된 빔 형성기를 위한 프로세서 시스템은 그림 4-11과 같다.

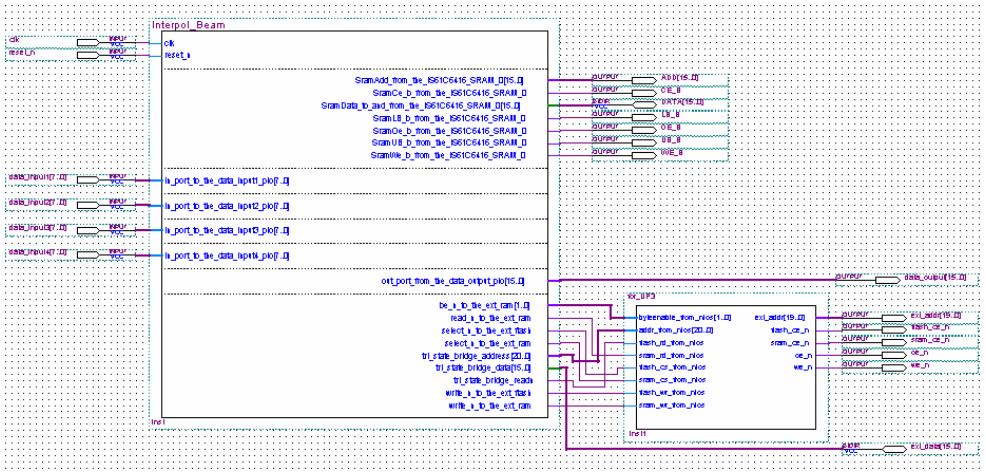


그림 4-11. 필터를 갖는 Delay-and-Sum 빔 형성기를 위한 시스템

Fig. 4-11. A System of Filtered Delay-and-Sum Beamformer.

Flow Status	Successful - Thu Oct 20 00:23:25 2005
Quartus II Version	5.0 Build 148 04/26/2005 SJ Full Version
Revision Name	beamformer
Top-level Entity Name	beamformer
Family	Cyclone
Device	EP1C6Q240C8
Timing Models	Final
Met timing requirements	No
Total logic elements	3,649 / 5,980 (61 %)
Total pins	91 / 185 (49 %)
Total virtual pins	0
Total memory bits	51,008 / 92,160 (55 %)
Total PLLs	0 / 2 (0 %)

그림 4-12. 설계된 시스템의 컴파일 정보

Fig. 4-12. Flow Summary of Designed System.

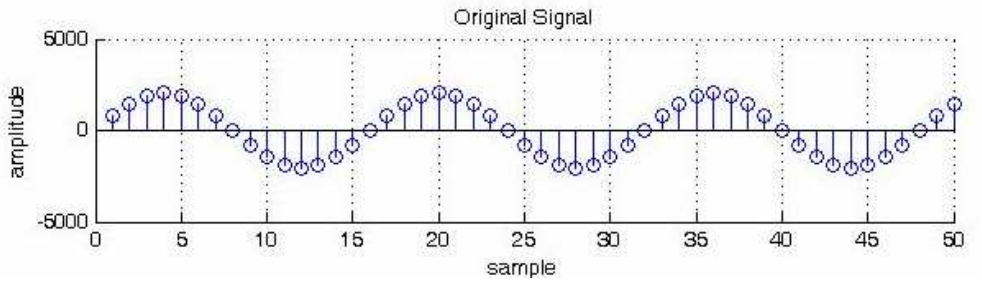
그림 4-12에 설계된 시스템의 컴파일 정보가 나타나 있다. Cyclone EP1C6Q240C8 칩 상에서 전체 LE 3,649/5,980 을 사용하며 61%를 차지하였으며, 메모리는 51,009 / 92,160 을 차지하며 55%를 사용하였다.

제 4-2 절 시뮬레이션 및 시스템 적용 결과 분석

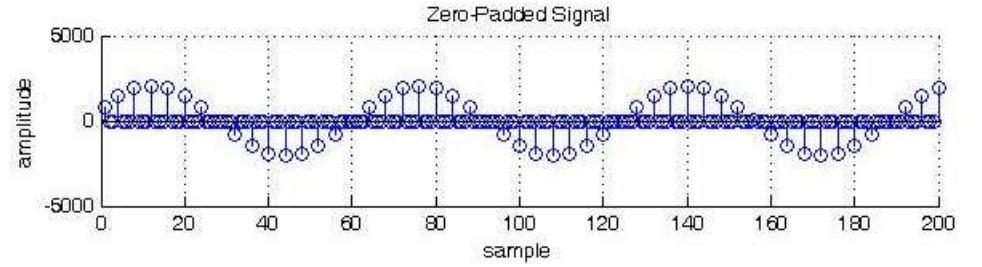
본 논문에서 설계한 빔 형성기는 크게 두 부분으로 구분된다. Zero-Padding 보간법과 Delay-and-Sum 처리 과정이다. 이 과정의 알고리즘을 앞에서 설계한 시스템에 적용시켜 전체 시스템을 설계하였다. 본 절에서는 설계된 시스템을 세 가지 방법으로 분석하였다. 첫째는 MATLAB을 통해 시뮬레이션 결과를 미리 확인한다, 다음으로 TI사의 TMS320C6711 DSP Processor를 통해 결과를 확인하고, 끝으로 본 논문에서 설계된 시스템에 적용시켜 그 결과를 비교, 분석하였다.

4-2-1. Zero-Padding 보간법에 대한 결과 분석

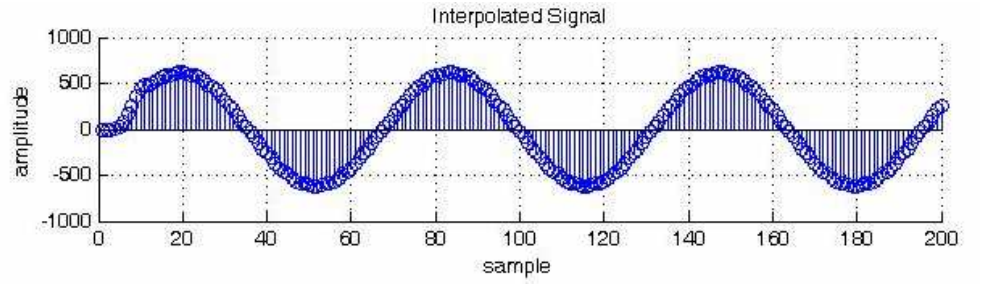
Zero-Padding 보간법이 제대로 수행되는지 알아보기 위해 우선 사인파를 생성하였다. 사인파는 샘플링 주파수가 8kHz이고 생성된 신호는 500Hz의 특성을 가진다. 이 신호는 20,000개의 샘플, 즉 2.5초의 길이를 가진다. 먼저 MATLAB을 통한 시뮬레이션 결과는 그림 4-13과 같다. 생성된 신호가 (a)에 나타나 있다. 결과 확인을 위해 50개의 데이터 샘플만 잘라서 확인하였다. (b)에는 4배로 보간하기 위해 데이터 샘플들 사이에 3개씩의 '0'이 배치된 상태를 보여준다. 데이터 개수가 200



(a)



(b)



(c)

그림 4-13. Zero-Padding 보간된 사인 신호의 시뮬레이션 결과,
(a) 원래 신호, (b) Zero-Padding 된 신호 (c) 보간된 신호

Fig. 4-13. Simulation Result of Zero-Padding Interpolated Sine Signal,
(a) Original Signal, (b) Zero-Padded Signal, (c) Interpolated Signal.

개로 늘어났음을 알 수 있다. (c)에서는 필터를 통과한 이후 보간된 신호를 확인할 수 있다. 원신호와 비교하여 데이터의 샘플수는 50개에서 200개로 늘어났다. 즉, 샘플주기가 1/4로 줄었다는 것은 샘플링 주파수가 4배로 늘어났다는 것을 의미한다. 파형을 통해 알 수 있지만, 원래 신호와 비교하여 신호의 특성 주파수가 변하지 않았음을 확인할 수 있다. 결국, 보간이 제대로 수행되었음을 확인하였다. 그러나, 보간된 신호의 크기가 원래 신호에 비해 많이 줄어든 것을 알 수가 있다. 이 것은 사용된 필터의 특성에 의한 결과로 그 필터의 특성은 그림 4-14에 나타내었다.

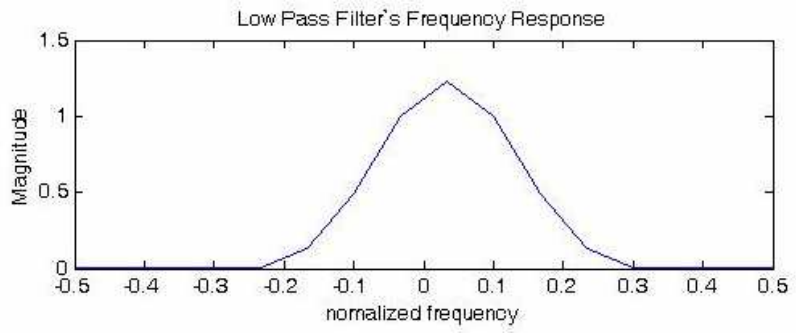
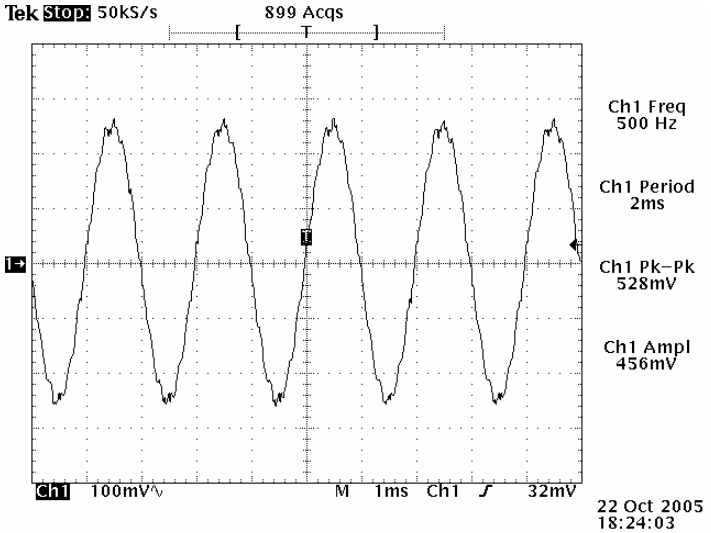


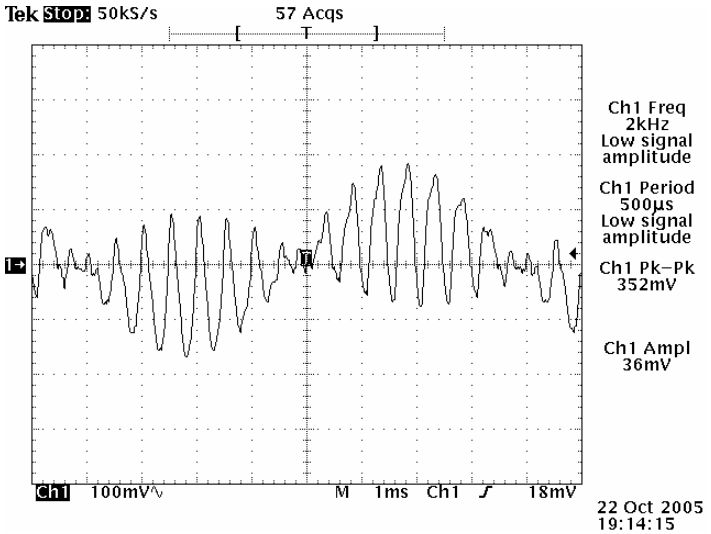
그림 4-14. 보간 필터의 주파수 응답

Fig. 4-14. Interpolation Filter's Frequency Response.

다음은 똑 같은 신호에 대해서 TI사의 TMS320C6711 DSP 프로세서를 사용하여 실제 처리를 하여 오실로스코프로 확인하였다.



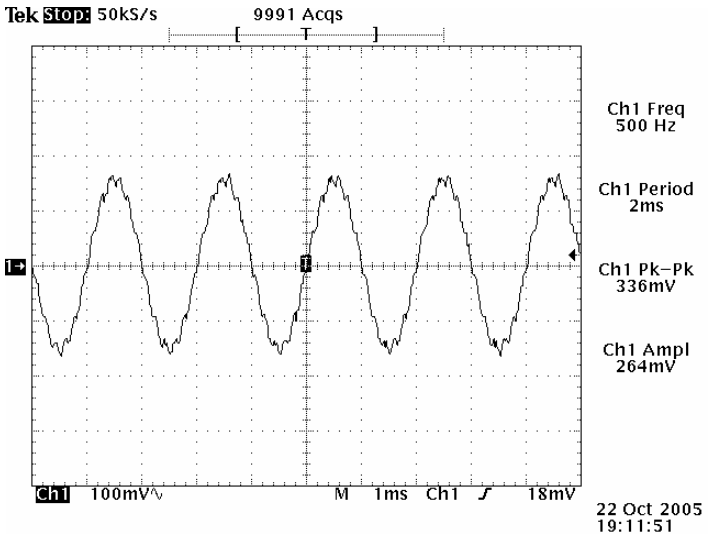
(a)



(b)

그림 4-15. Zero-Padding 보간된 사인 신호의 DSP 처리 결과,
(a) 원래 신호, (b) Zero-Padding 된 신호, (c) 보간된 신호.

Fig. 4-15. DSP Result of Zero-Padding Interpolated Sine Signal,
(a) Original Signal, (b) Zero-Padded Signal, (c) Interpolated Signal.

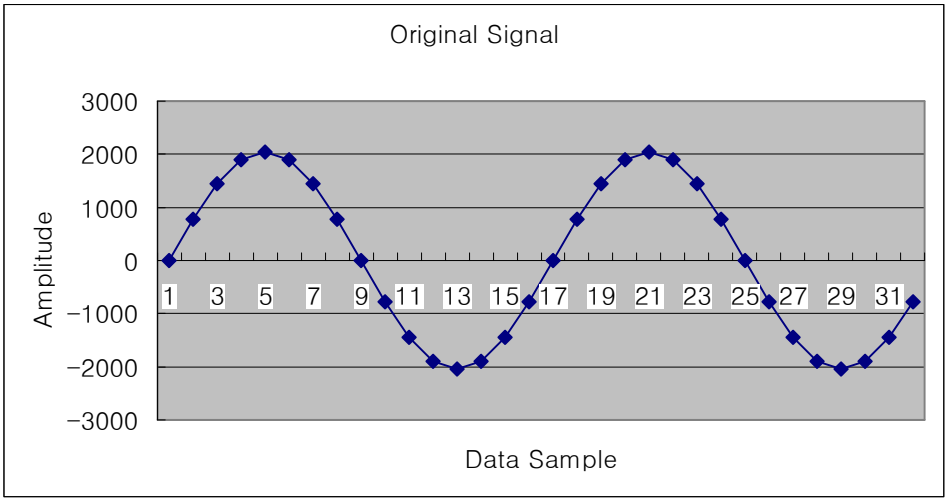


(c)

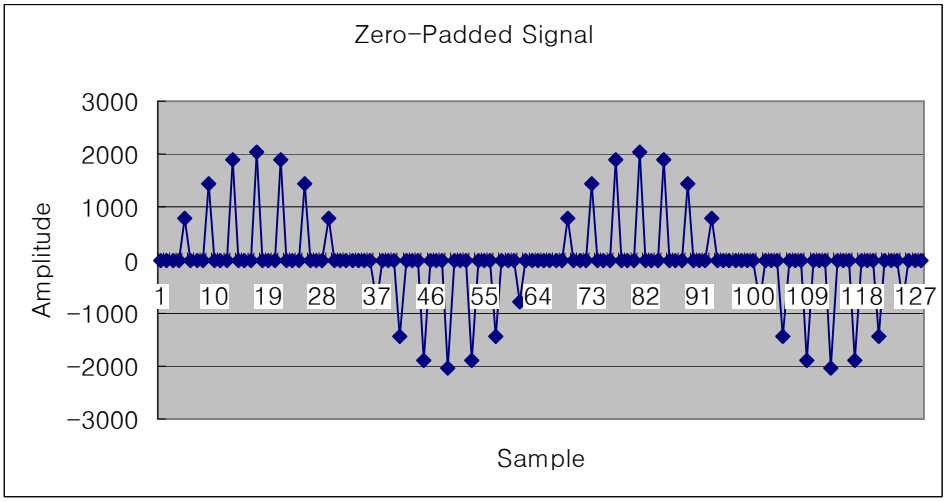
그림 4-15. Zero-Padding 보간된 사인 신호의 DSP 처리 결과,
(a) 원래 신호, (b) Zero-Padding 된 신호, (c) 보간된 신호

Fig. 4-15. DSP Result of Zero-Padding Interpolated Sine Signal,
(a) Original Signal, (b) Zero-Padded Signal, (c) Interpolated Signal.

그림을 통해 알 수 있듯이 시뮬레이션에 비교하여 진폭의 크기가 감소하였지만, 원 데이터 사이의 값들을 제대로 보간하고 있음을 확인할 수 있다. 끝으로 본 논문에서 설계된 시스템에서의 Zero-Padding 보간법의 결과가 그림 그림 4-16에 있다.



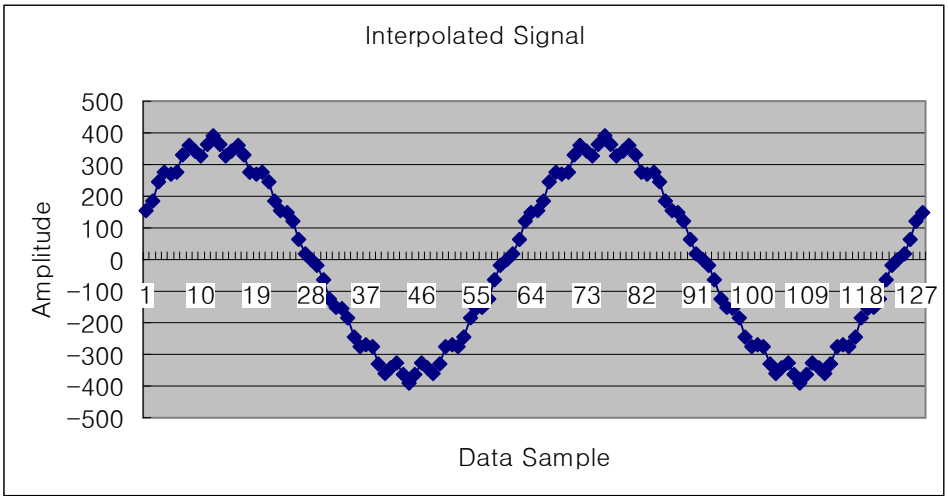
(a)



(b)

그림 4-16. Zero-Padding 보간된 사인 신호의 설계된 시스템에서의 처리 결과, (a) 원래 신호, (b) Zero-Padding 된 신호, (c) 보간된 신호

Fig. 4-16. Processing Result of Zero-Padding Interpolated Sine Signal on Designed System, (a) Original Signal, (b) Zero-Padded Signal, (c) Interpolated Signal.



(c)

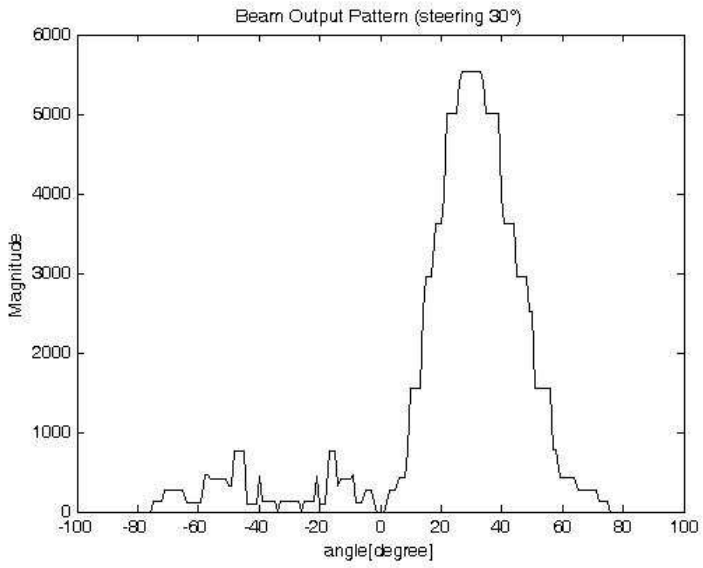
그림 4-16. Zero-Padding 보간된 사인 신호의 설계된 시스템에서의 처리 결과, (a) 원래 신호, (b) Zero-Padding 된 신호, (c) 보간된 신호

Fig. 4-16. Processing Result of Zero-Padding Interpolated Sine Signal on Designed System, (a) Original Signal, (b) Zero-Padded Signal, (c) Interpolated Signal.

설계된 시스템에 사용된 보드는 주변 장치로 코덱이나 실제 데이터의 입출력이 가능한 장치가 장착되어 있지 않다. 그래서 시스템 Running을 통해 획득한 데이터를 수집하여 엑셀로 그린 결과이다. 그림을 통해 확인할 수 있듯 '0'의 값이 깔리는 과정과 필터 통과 후 보간된 값들이 제대로 출력되고 있음을 확인할 수 있다.

4-2-2. 설계된 Delay-and-Sum 빔 형성기의 결과 분석

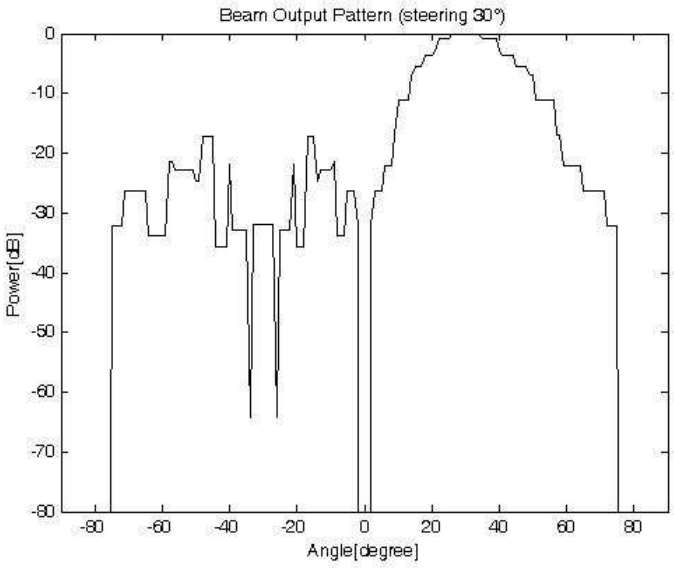
앞 절에서 검증된 보간법에 따라 해상도가 높아진 신호를 이용하여 설계된 빔 형성기에 적용하여 그 결과를 확인하였다. 사용된 신호는 마이크로폰 배열의 중심으로부터 30° 방향에서 재생되며, 서로 5 샘플의 지연을 가지고 4채널로 입력되는 신호를 사용하였다. 그 신호는 각각 보간법을 거친 후 설계된 빔 형성기 시스템으로 입력되어 최종 출력 빔을 형성하게 된다. 출력 빔 패턴의 결과를 예상하기 위하여 먼저 MATLAB을 이용하여 시뮬레이션을 한 결과가 그림 4-17에 있다.



(a)

그림 4-17. 출력 빔 패턴 (4채널, 지향각 30°) (계속)

Fig. 4-17. Output Beam Pattern (4 channel, steering 30°) (continue)

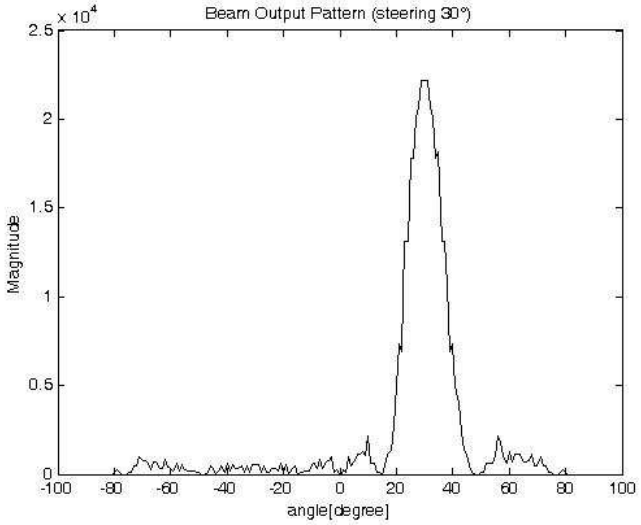


(b)

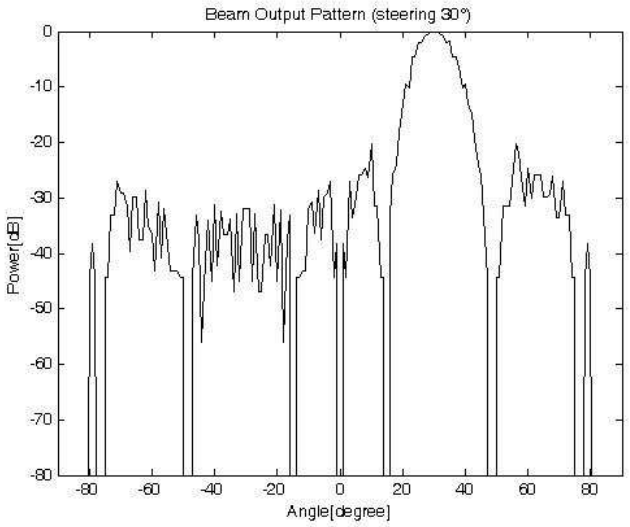
그림 4-17. 출력 빔 패턴 (4채널, 지향각 30°)
(a) 파워 값, (b) 파워 (dB값)

Fig. 4-17. Output Beam Pattern (4 channel, steering 30°)
(a) Power Value, (b) Power (dB scale).

출력 파워를 확인한 결과 30°에서 이득이 가장 집중되는 것을 알 수 있다. 그러나 그림 4-17의 (b)에서 dB 스케일로 변환하여 확인한 결과 30° 에서 가장 높은 이득을 보이긴 하지만, 다른 부엽들과의 이득 차이가 20dB정도로 성능이 떨어져 원활한 빔 형성 수행이 어렵다는 것을 알 수 있다. 이러한 이유는 입력단의 채널 수 부족 및 시간 축 해상도가 원인이라 할 수 있다. 이는 앞서 설명된 보간법을 이용하여 해결할 수 있다. 그에 따라 8채널로 신호를 입력 받아서 그 결과를 확인하여 그림 4-18에 나타내었다.



(a)



(b)

그림 4-18. 출력 빔 패턴 (8채널, 지향각 30°)

(a) 파워 값, (b) 파워 (dB값)

Fig. 4-18. Output Beam Pattern (8 channel, steering 30°)

(a) Power Value, (b) Power (dB scale).

채널 수를 4채널에서 8채널로 늘린 결과 주 빔과 부엽들의 이득차이가 30 dB정도 이상 차이를 보임을 알 수 있다. 즉, 채널 수의 확장을 위해 빔의 이득에 대한 성능의 향상을 가져올 수 있다. 그에 따라 본 논문에서는 보유하고 있는 Nios-II 보드의 한계로 인하여 프로토타입의 성능 검증으로써 4채널만을 사용하여 시스템에 적용하였다. DSP 프로세서를 이용한 결과가 그림 4-19에 있다.

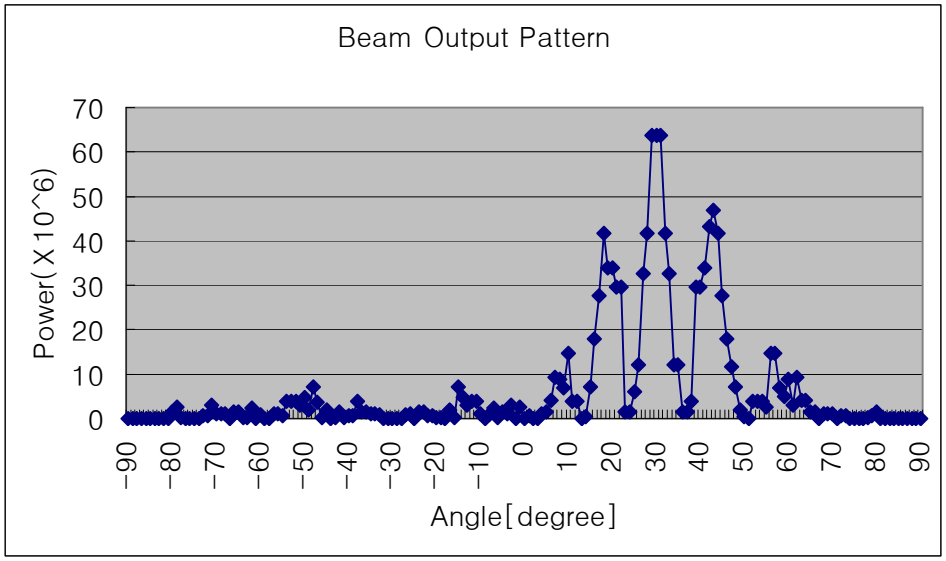


그림 4-19. DSP 프로세서를 이용한 출력 빔 패턴

Fig. 4-19. Output Beam Pattern using DSP Processor.

데시벨(dB) 스케일로의 변환은 연산량의 문제로 생략하였으며, 파워 값만을 통해 빔 출력을 확인하였다. 지향각인 30° 부근에서 이득이 가장 집중되고 있으나 10° ~ 50° 영역에서 주 빔을 제외한 부엽의 이득이 다소 높은 것으로 나타난다. 이는 MATLAB과 실제 하드웨어의 연산 시에 해상도의 차이로 인해 발생하는 결과라고 할 수 있다. 마지막으로

본 논문에서 설계한 Nios-II Processor를 이용한 결과는 그림 4-20에 나타내었다.

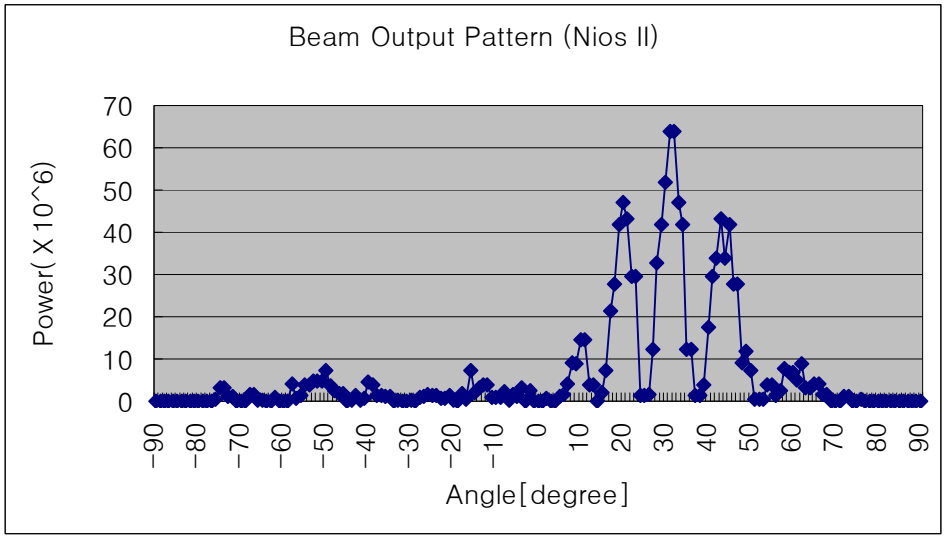


그림 4-20. Nios-II 프로세서를 이용한 출력 빔 패턴

Fig. 4-20. Output Beam Patter using Nios-II Processor.

빔 출력의 패턴은 DSP 프로세서로 처리했을 경우와 비슷한 것을 확인할 수 있다. 각각의 부엽들에서 약간의 이득차이를 보이긴 하지만, 이들은 무시될 수 있는 값들이다. 즉, 필터의 과정이나 시간 지연을 구하기 위한 과정에서의 곱셈 연산 등의 처리에서 해상도나 속도가 떨어지지 않음을 확인하였다.

제 5 장 결 론

차량 주행 중 사용자의 안전을 위해 도입된 핸즈프리는 차량 내, 외의 잡음으로 인해 원활한 통신에 장애가 따르는 문제점을 가지고 있다. 이러한 통신 장애를 극복하기 위해 원하는 음성만을 픽업하여 통신을 하는 기법으로 빔 형성 기법을 적용하였다. 빔 형성기는 하드웨어를 위해 고전적이면서도 그 접근이 용이한 Delay-and-Sum 빔 형성기를 선택하였다. 코덱을 통해 입력 받는 데이터만으로는 신호의 해상도가 떨어지기 때문에, 빔 형성 절차에 앞서 보간법을 적용하였다.

본 논문에서 설계된 보간 필터를 갖는 Delay-and-Sum 빔 형성기는 FPGA를 이용하여 실제 제작되었다. 기존의 SHARC DSP만을 이용하여 설계된 빔 형성기의 경우 충분한 데이터의 확보가 힘들어 많은 메모리의 확장과 파이프라인 형태로 DSP간 연결이 필요하였다. 또한 FPGA만을 통해 빔 형성기를 설계한 경우 데이터 입. 출력을 제어하는 부가적인 프로세서 개념의 하드웨어를 부착하였는데, 이 경우 설계 사양에 맞춰 하드웨어를 제작하게 되므로 많은 시간을 투자해야 한다는 것과 비용이 많이 들게 된다. 특히 사양에서 벗어난 빈 공간을 감수해야 하며, 그에 따라 설계 목적 외에는 사용이 힘들다는 단점이 있었다. 즉, 시스템의 업그레이드가 필요할 경우 하드웨어를 새로 설계, 제작해야 하는 결점이 있다.

따라서 본 논문에서는 이러한 결점들을 극복하고자 ALTERA 사의 Nios-II Processor를 이용하여 빔 형성기의 입. 출력을 제어하고 연산의 성능을 향상시켰다. 실제로 Nios-II 는 32비트 RISC구조 임베디드 프로세서로써 200Mips의 속도를 제공한다. 이 Nios-II를 통해 설계된 본 논문의 빔 형성기는 'Avalon Switch Fabric'을 통한 버스 관리를 통

해 일반 프로세서보다 향상된 대역폭을 확보하였으며, 커스텀 인스트럭션으로 프로세서에 DMA와 SRAM을 추가함으로써 소프트웨어상으로 실행될 때보다 빠른 수행 결과를 얻는다. 소프트웨어적인 결과로써 빔 형성 4채널의 마이크로폰을 통해 입력 받은 신호는 Zero-padding 보간법을 사용한 결과 4배의 샘플링 주파수 향상과 2배의 비트 해상도가 향상되었다. 이 신호로써 빔 형성 알고리즘을 통해 얻은 출력 결과는 부동소수점 연산을 통해 DSP로 처리한 결과에 비해 해상도 측면에서 떨어지지 않는 결과를 확인하였다. 이렇게 설계된 빔 형성기는 LE가 3,649/5,980 만큼 사용되었다. 이는 더 많은 응용분야의 적용이 가능함을 말해주며, 더욱 사양이 높은 Cyclon-II, 혹은 Stratix 계열의 Nios-II 보드를 사용할 경우 더욱 많은 채널의 추가와 응용 알고리즘을 적용할 수 있음을 말해준다.

향후 과제로써는 우선 충분히 많은 수의 채널을 추가하여 빔 출력의 안정성을 높이고, 또한 하드웨어 가속화 기능을 사용하여 처리 속도를 더욱 향상시켜야 할 것이다. 그리고 궁극적으로 보드에 코덱 등의 외부 장치를 확대하여 실제 차량 내에서 실험을 통해 실시간으로 빔 형성기의 성능 검증이 이루어져야 할 것이다.

참고문헌

[1] Korea Afis. Co. *http://www.afis.co.kr/afis/info/hwhy.htm*

[2] Seungil Kim, Optimum Beamformer Using Correlated Interferences and Its Application to Telematics System, Ph.D dissertation, Yonsei University, 2004..

[3] Richard O. Nielson, *Sonar Signal Processing*, Artech House, pp.22-81, 1991.

[4] 김상균, 김기영, 오선택, 김형곤, "FPGA를 적용한 능동 소나 신호처리 시스템의 시간영역 빔형성기 설계," *제20회 수중음향 학술발표대회*, pp.67-70, Oct. 2005.

[5] Russell J. Pertersen and Brad L. Huntchings, "An assessment of the suitability of FPGA-based systems for use in Digital Signal processing," *Proceeding of the 5th International Workshop on Field-Programmable Logic and Applications*, pp.293-302, September, 1995.

[6] Paul Graham and Brent Nelson, "FPGA-Based Sonar Processing," *Proceedings of the 1998 ACM/SIGDA sixth International Symposium on Field Programmable Gate Arrays (Monterey, CA, USA, 1998)*, pp.201-208.

[7] B. Widerow, P.E. Mantey, L.J. Griffiths, and B. B. Goode, "Adaptive Antenna Systems," *Proc. IEEE*, vol. 55, pp.2143-2159, Dec. 1967.

[8] Frank Engel, Peter Mumford, Kevin Parkinson, Chris Rizos, and Gernot Heiser, "An open GNSS Receiver platform architecture," *Journal of Global Positioning Systems*, vol.3, no.1-2, pp.63-69, 2004.

[9] Jason Cong, Yiping Fan, Guoling Han, Ashok jagannathan, Glenn Reinman, and Zhiru Zhang, "Instruction set extension with shadow registers for configurable processors," *13th ACM International Symposium on Field-Programmable Gate Arrays*, pp.99-105, Feb 2005.

[10] Paul Graham and Brent Nelson, "FPGAs and DSPs for Sonar Processing – Inner Loop Computations," *Techmical. Report CCL-1998-GN- 1*, Configurable. Computation Laboratory, Electrical and. Computer Engineering Department.

[11] Simon Haykin, *Adaptive Filter Theory – 3rd edition*, Prentice hall, NJ, 1996.

[12] J. Capson, "High Resolution Frequency-Wavenumber Spectrum Analysis," *Proc. of IEEE*, vol. 57, pp. 1408-1418, Aug. 1969.

[13] Peter Yiannacouras, Jonathan Rose, and J. Gregory Steffan, "The microarchitecture of FPGA-based soft processors," *Proceedings of the 2005 International Conference on Compilers, Architectures and Synthesis for Embedded Systems*, pp.202-212, 2005.

[14] U. Meyer-Baese, *Digital Signal Processing with Field Programmable Gate Arrays*, Springer, 2001.

[15] Rulph Chassaing, *DSP Applications Using C and the TMS320C6x DSK*, John Wiley & Sons, Inc., 2002.

[16] Nios II Processor Reference Handbook, *http://www.altera.com/literature/lit-nio2.jsp*

[17] Nios II Software Developer's Handbook, *http://www.altera.com/literature/lit-nio2.jsp*

[18] ESDK Reference Manual, *http://www.slscorp.com*

[19] Nios Development Board Reference Manual, Cyclone Edition , *http://www.altera.com/literature/lit-nio2.jsp*

[20] TMS320C6000 Peripherals Reference Guide, *http://www.ti.com/*