Basic Camera Operation for WP:

Name of Experiment: Basic Camera Operation	Exp No: WP5
Background: Student should have a basic knowledge of C#.	
Summary: After going through this experiment, student should understand how to access the camera hardware through the code and could able to learn the basic operation on camera.	
Learning Objective: To explain the working of camera hardware.	
Target Platforms: This application is tested on Windows Phone Emulator and Windows Phone (Lumia 800).	

Procedure:

Step 1. Please repeat steps[1-6] as in experiment no WP1.

Step2. . Go to Stack Panel->Text Block->Name= “Experiment Title” and Text=“Exp No: WP5” and set Text Alignment as “RIGHT”.

Step 3. Do the similar task, go to Stack Panel->Text Block->Name = “Application Title and Text= “Basic Camera Operation”.[Refer MainPage.xaml in Source Code section]

Step 4. Now, drag and drop two buttons from Toolbox in the content panel make them align. Set the Button content as “Photo” and “Show”. Drop one Image box on the content panel and name it as “Image1”. This Image box will contain the picture that is captured by camera hardware device and will shown to the user.[Refer fig. no 1]

Step5. Add click event handler on the two buttons “Photo” and “Show”.

Step6. Initialise the “PhotoChooserTask _myphoto = new PhotoChooserTask()” constructor and “CameraCaptureTask _mycamera = new CameraCaptureTask()” constructor before the main class component initialise.[Refer Source code section]

Step7. Add _mycamera task event delegate to Main Page “_mycamera.Completed+=new EventHandler<PhotoResult>(_mycamera_Completed)” and _myphoto event delegate to Main Page “_myphoto.Completed+=new EventHandler<PhotoResult>(_myphoto_Completed)”.[Refer Source Code Section]

Step8. Save all the changes that we have made by simply clicking the Alt+S or Save option from the above pallete.

Step9. Go to the Debug option from the menu and click on the start Debug.

Step10. An emulator will start after few min and the Home Screen page is shown to you.[Refer fig. no 2]

Step11. By this way, a Basic Camera Operation program can be deployed on the WP7.

Step12. To deploy the Basic Camera Operation program on Windows Phone device [Refer Chapter 2]

Basic Camera Operation for WP:

Source Code	Comments

Main Page.xaml
`<!--TitlePanel contains the name of the application and page title-->`
`<StackPanel x:Name="TitlePanel" Grid.Row="0" Margin="12,17,0,28">`
`<TextBlock x:Name="ExperimentTitle" Text="Exp No:WP5" TextAlignment="Right" Style="{StaticResource PhoneTextNormalStyle}"/>`
`<TextBlock x:Name="ApplicationTitle" Text="Basic Camera Operation" Style="{StaticResource PhoneTextNormalStyle}"/>`
`<TextBlock x:Name="PageTitle" Text="Welcome Page" Margin="9,-7,0,0" Style="{StaticResource PhoneTextTitle1Style}"/>`
`</StackPanel>`
 	
`<!--ContentPanel - place additional content here-->`
`<Grid x:Name="ContentPanel" Grid.Row="1" Margin="12,0,12,0">`
`<Grid.RowDefinitions>`
`<RowDefinition Height="101*" />`
`<RowDefinition Height="506*" />`
`</Grid.RowDefinitions>`
`<Button x:Name="Photo" Content="Photo" Height="72" HorizontalAlignment="Left" Margin="12,29,0,0" VerticalAlignment="Top" Width="160" />`
`<Image Grid.Row="1" Height="306" HorizontalAlignment="Right" Margin="0,68,46,0" Name="Image1" Stretch="Fill" VerticalAlignment="Top" Width="356" />`
`<Button Content="Show" Height="72" HorizontalAlignment="Left" Margin="227,29,0,0" Name="cmd_show" VerticalAlignment="Top" Width="160" />`
`</Grid>`
`</Grid>`
	← Name= “ExperimentTitle” Text= “Exp No:WP3”
	
	←Name= “ApplicationTitle” Text= “Basic Camera Opertaion”
	
	← Name= “PageTitle” Text= “Welcome Page”
	
	← Button Name= “Photo” Content= “Photo”
	
	← Image Box Name= “Image1”
	
	← Button Name= “cmd_show” Content= “Show”

Main Page.xaml.cs
`using System;`
`using System.Windows;`
`using Microsoft.Phone.Controls;`
`using Microsoft.Phone.Tasks;`
`using System.Windows.Media.Imaging;`
`using System.Windows.Media;`
 	
`namespace Camera_Test`
	← Task Event
	← Bitmap images

Basic Camera Operation for WP:

```
{
    public partial class MainPage :
PhoneApplicationPage
    {
        CameraCaptureTask _mycamera = new
CameraCaptureTask();
        PhotoChooserTask _myphoto = new
PhotoChooserTask();
        // Constructor
        public MainPage()
        {
            InitializeComponent();
            _mycamera.Completed+=new
EventHandler<PhotoResult>(_mycamera_Completed);
            Photo.Click+=new
RoutedEventHandler(Photo_Click);
            _myphoto.Completed+=new
EventHandler<PhotoResult>(_myphoto_Completed);
            cmd_show.Click+=new
RoutedEventHandler(cmd_show_Click);



        }



        void _mycamera_Completed(object sender,
PhotoResult e)
        {
            if (e.TaskResult == TaskResult.OK)
{
                Image1.Source = new
BitmapImage(new Uri(e.OriginalFileName));
                BitmapImage img = new
BitmapImage();
                img.SetSource(e.ChosenPhoto);
                Image1.Source = img;
            }
         }

        void Photo_Click(object sender,
RoutedEventArgs e) {
            _mycamera.Show();
        }
        void _myphoto_Completed(object
sender,PhotoResult e) {
            if (e.TaskResult == TaskResult.OK)
            { ImageBrush myimagebrush = new
ImageBrush();
            myimagebrush.ImageSource = new
BitmapImage(new Uri(e.OriginalFileName,
UriKind.Relative));
            Image1.Source = new BitmapImage(new
Uri(e.OriginalFileName, UriKind.Relative));
            BitmapImage img = new
BitmapImage();
            img.SetSource(e.ChosenPhoto);
            Image1.Source = img;
            }
        }
        void cmd_show_Click(object sender,
RoutedEventArgs e)
        {
```

← CameraCaptureTask();
← PhotoChooserTask();

←CameraCaptureTask completed Event Handler.

←PhotoChooserTaskCompleted Event Handler.

← Body for Camera Capture task Completed Event Handler

← Button Photo Click event Handler

← Body for Photo Chooser task Completed Event Handler

← Button Show event handler

```
            _myphoto.Show();
        }
    }
}
```

Screenshots

fig. no 1 Layout for UI design

fig. no 2 Welcome Page (output on emulator)

Fig. no 3 Camera Hardware(output on emulator)

fig. no 4 Capture event fired(saving image)

Basic Camera Operation for WP:

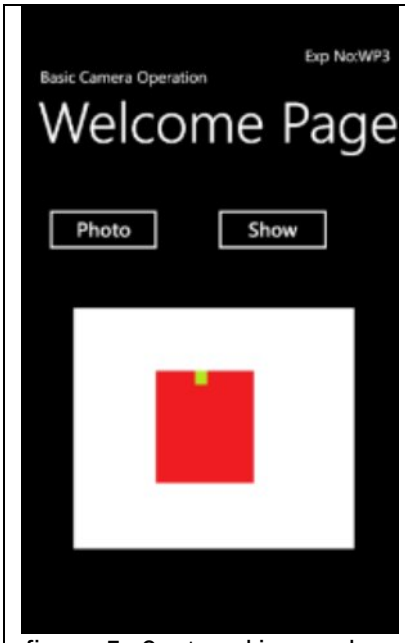

fig. no 5 Captured image shown to User

fig. no 6 Choose picture from camera roll(Show button)

fig. no 7 After selection of an Image

Observations:

After going through this experiment, developer can analyse that camera hardware can be easily controlled by the soft code.