Jurnal Teknologi

Full Paper

DYNAMIC POWER DISSIPATION FORMULATION FOR APPLICATION IN DYNAMIC PROGRAMMING BUFFER INSERTION ALGORITHM

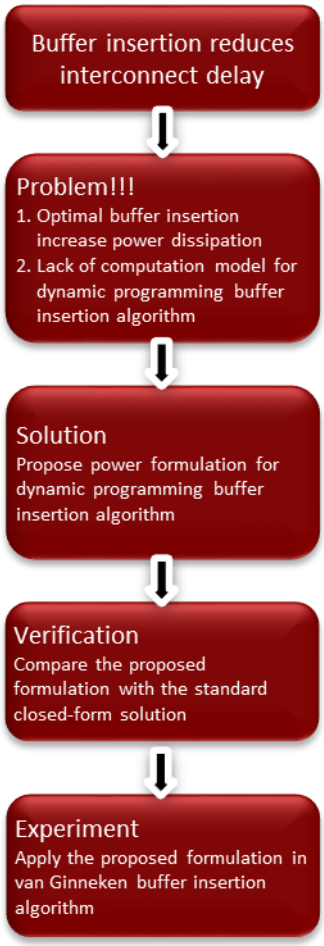
Chessda Uttraphan[a,b*], Nasir Shaikh-Husin[b], M. Khalil-Hani[b]

[a]Embedded Computing Systems (EmbCoS) Research Focus Group, FKEE, Universiti Tun Hussein Onn Malaysia
[b]VeCAD Research Laboratory, Universiti Teknologi Malaysia

Article history
Received
1 July 2015
Received in revised form
10 August 2016
Accepted
18 October 2016

Corresponding author
chessda@uthm.edu.my

Graphical abstract



Abstract

Buffer insertion is a very effective technique to reduce propagation delay in nano-metre VLSI interconnects. There are two techniques for buffer insertion which are: (1) closed-form solution and (2) dynamic programming. Buffer insertion algorithm using dynamic programming is more useful than the closed-form solution as it allows the use of multiple buffer types and it can be used in tree structured interconnects. As design dimension shrinks, more buffers are needed to improve timing performance. However, the buffer itself consumes power and it has been shown that power dissipation of buffers is significant. Although there are many buffer insertion algorithms that were able to optimize propagation delay with power constraint, most of them used the closed-form solution. Hence, in this paper, we present a formulation to compute dynamic power dissipation of buffers for application in dynamic programming buffer insertion algorithm. The proposed formulation allows dynamic power dissipation of buffers to be computed incrementally. The technique is validated by comparing the formulation with the standard closed-form dynamic power equation. The advantage of the proposed formulation is demonstrated through a series of experiments where it is applied in van Ginneken's algorithm. The results show that the output of the proposed formulation is consistent with the standard closed-form formulation. Furthermore, it also suggests that the proposed formulation is able to compute dynamic power dissipation for buffer insertion algorithm with multiple buffer types.

Keywords: Dynamic programming, buffer insertion, CMOS power dissipation

Abstrak

Sisipan penimbal adalah satu teknik yang sangat efektif untuk mengurangkan lengah dalam penghubung VLSI nano-meter. Terdapat dua teknik untuk sisipan penimbal iaitu (1) penyelesaian format-tertutup dan (2) pemprograman dinamik. Sisipan penimbal dengan menggunakan pemprograman dinamik lebih berguna berbanding dengan penyelesaian format-tertutup kerana ia membenarkan penggunaan pelbagai jenis penimbal dan ia juga boleh digunakan dalam penghubung berstruktur ranting. Dengan penyusutan dimensi rekebentuk, semakin banyak penimbal diperlukan untuk meningkatkan prestasi pemasaan. Namun begitu, penimbal itu sendiri menggunakan kuasa dan kajian telah menunjukkan bahawa pelepasan kuasa dari penimbal adalah cukup ketara. Walaupun terdapat banyak algoritma untuk sisipan penimbal yang berupaya untuk mengoptimumkan lengah perambatan dengan penghadan kuasa, kebanyakannya menggunakan penyelesaian format-tertutup. Oleh demikian, dalam kertas kerja ini, kami mempamerkan formulasi untuk menghitung pelepasan kuasa dinamik oleh penimbal untuk digunakan dalam algoritma sisipan penimbal pemprograman dinamik. Formulasi yang dicadangkan ini membolehkan pelepasan kuasa dinamik oleh penimbal dapat dihitung secara tambahan. Teknik ini disahkan dengan membandingkan formulasi cadangan dengan persamaan penyelesaian format-tertutup piawai. Kelebihan fomulasi cadangan didemonstrasikan melalui beberapa siri eksperimen di mana ia diaplikasikan dalam algorithma van Ginneken. Keputusan menunjukkan bahawa keluaran dari formulasi

cadangan adalah konsisten dengan formulasi format-tertutup piawai. Tambahan pula, ia juga mencadangkan bahawa formulasi cadangan berkeupayaan untuk menghitung pelepasan kuasa untuk algorithma sisipan penimbal dengan pelbagai jenis penimbal.

Kata kunci: Pemprograman dinamik; sisipan penimbal; pelepasan kuasa CMOS

1.0 INTRODUCTION

In nano-metre VLSI designs, the interconnect delay has become more significant compared to gate delay [1]. One of the effective techniques to reduce interconnect delay is by inserting a repeater (buffer) to reconstruct the signal along the interconnect tree [1]. Buffer insertion algorithm that finds the best locations for buffer insertion was proposed by van Ginneken [2]. The algorithm is based on dynamic programming where the candidate solutions of capacitance and delay are computed incrementally, from sink to the source. The optimum solution is obtained at the source. Recently, design dimension in VLSI has been reduced. As a result, more buffers are needed to improve timing performance. However, buffer itself consumes power and it has been shown that power dissipation overhead of inserted buffers is significantly high [3]. Many methodologies to optimize propagation delay with power constraint have been proposed such as in [3 – 5] but none of them can be integrated into buffer insertion algorithm that is based on dynamic programming technique [6] as they use closed-form solution. The available techniques that calculate power dissipation iteratively for dynamic programming buffer insertion algorithm were proposed by [7 – 9]. However, their methods do not reflect the actual power calculation. For example, in [7] and [8] proposals, the power is represented by a cost function, which is a capacitance as it is proportional to dynamic power dissipation. [9] also uses capacitance to represent dynamic power, but with additional leakage power, which is represented by buffer area. Hence, we propose a formulation that can compute power consumption of buffers incrementally based on dynamic programming framework. This formulation enables any van Ginneken style buffer insertion algorithm to also consider power consumption of buffers.

2.0 METHODOLOGY

2.1 Dynamic Programming Buffer Insertion

Dynamic programming [6, 10] is essentially a divide-and-conquer method where a complex problem is solved by combining the solutions of the sub-problems. This technique can be summarized as follows: (1) dividing the problem into smaller sub-problems, (2) solving the smaller sub-problems optimally and (3) combining the optimal solutions of the sub-problems to get a solution to the original problem. The advantages of dynamic programming technique to find optimal buffer insertions over the closed-form solution is that it can be used to optimize multi-pin nets and can handle different buffer types.

In buffer insertion algorithm, the interconnect wire is divided into equal segments as shown in Figure 1. The label *source* is the source of the signal and the label *sink* is the destination of the signal. Each wire segment is modelled as π-model RC circuit as shown in Figure 2a while the buffer model is shown in Figure 2b. The label c_w and r_w are the capacitance and resistance per wire segment, while r_b, c_b and d_b are the output resistance, input capacitance and intrinsic delay of the buffer respectively.

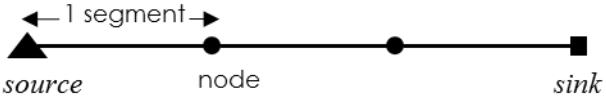


Figure 1 An interconnect wire divided into three segments

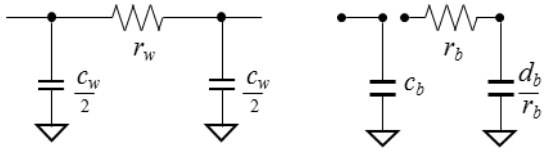


Figure 2 (a) Wire segment model (b) Buffer model

van Ginneken was the first to utilize the dynamic programming technique in buffer insertion algorithm [2]. The goal of the algorithm is to determine the best location of buffers on a given interconnect (at the node between each segment) in order to optimize the Elmore delay [11]. The delay is calculated for each segment starting from sink node toward the source (this is called upstream computation). The computation is characterized by two parameters which are downstream capacitance and downstream delay. This capacitance-delay (c,t) pair is called a candidate solution. In dynamic programming, this candidate solution is expanded toward the source by the following operations:

(1) Wire insertion: Propagate the candidate solution from node v to u by inserting a wire segment between v and u. If (c,t) is the candidate solution at node v, then the new candidate solution at node u is (c',t') pair given by

$$c' = c + c_w \quad (1a)$$

$$t' = t + r_w\left(\frac{c_w}{2} + c\right) \quad (1b)$$

(2) Buffer insertion: Insert a wire segment between v and u with a buffer at node v, and then add a new candidate into the solution set. If (c,t) is the candidate solution at node v, then the new candidate solution (c',t') at node u is given by

$$c' = c_w + c_b \quad (2a)$$

$$t' = r_w\left(\frac{c_w}{2} + c_b\right) + d_b + r_b c + t \quad (2b)$$

(3) In a tree structured interconnect, if a solution reaches a Steiner node, the candidate solution from the left branch $(c,t)_{left}$ is merged with the candidate solution from the right branch $(c,t)_{right}$. The merging solution (c',t') is given by

$$c' = c_{left} + c_{right} \quad (3a)$$

$$t' = \max(t_{left}, t_{right}) \quad (3b)$$

(4) When the candidate solution reaches the source node, the delay at source is computed with consideration for the source resistance, R_s as follows

$$t_{source} = t + cR_s \quad (4)$$

2.2 Power Dissipation in Buffered Path Interconnect

When more buffers are inserted in a long interconnect wire, the overall signal delay will be reduced. However, buffer itself consumes power and this implies that signal delay and power consumption of the interconnect move in opposite directions. Hence, buffer insertion algorithm should be able to handle power dissipation constraint [3, 12]. Power dissipation of the CMOS buffer arises from three sources summarized as follows [13]:

$$P_T = \alpha C V_{DD}^2 f + I_{sc} V_{DD} + I_{leakage} V_{DD} \quad (5)$$

The first term represents the switching power or dynamic power P_d, where f is the clock frequency, C is the total load capacitance and α is the switching factor. The second term is due to the direct-path short circuit current I_{sc}, which arises when both NMOS and PMOS transistors are simultaneously active, conducting current directly from power source to ground. The last term is the leakage power which arises from substrate injection and sub-threshold current effects. By assuming $C_L = c_b$, the closed-form solution for dynamic power consumption for an interconnect of length L with m number of inserted buffers is given as follows [5, 14]:

$$P_d = \alpha\left(c_w L + m\left(c_b + \frac{d_b}{r_b}\right)\right)V_{DD}^2 f \quad (6)$$

The leakage power and short circuit power are not considered in this work because they do not depend on wire capacitance. To include leakage and short circuit powers in the algorithm, one can pre-compute these powers in the buffer libraries.

2.3 Proposed Formulation

This section explains the procedure to compute the dynamic power consumption in buffers incrementally (dynamic programming) based on Eq. (6). According to van Ginneken algorithm, the operation for wire expansion from node v to node u can be performed as illustrated in Figure 3. Instead of (c,t) pair as in conventional algorithm, the buffer insertion algorithm with power constraint will have a candidate solution with three-tuple which are capacitance-delay-power or (c,t,p). Hence, if the wire is expanded from node v to node u, the new capacitance c' and delay t' for node u are computed using Eq. (1) and the dynamic power dissipation p' for node u is given by

$$p' = p \quad (7)$$

When a wire is expanded from node v to node u and a buffer is inserted at node v as shown in Figure 4, the new capacitance c' and delay t' for node u are computed using Eq. (2) and the dynamic power dissipation p' for node u is given by

$$p' = p + \alpha\left(\frac{d_b}{r_b} + c\right)V_{DD}^2 f \quad (8)$$

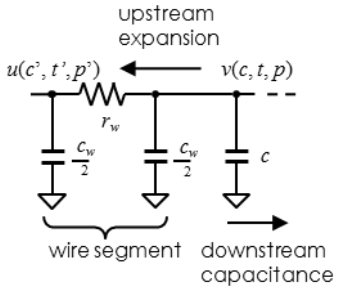


Figure 3 Wire expansion

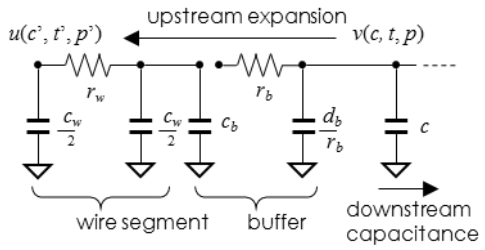


Figure 4 Wire expansion terminated by buffer

For the tree structured interconnect, when a solution from the right child $(c,t,p)_{right}$ and the left child $(c,t,p)_{left}$ meet at a Steiner node, the branch merging operation is performed where c' and t' are computed using Eq. (3) and p' is given by

$$p' = p_{right} + p_{left} \tag{9}$$

Finally when the solution reaches the source node, the total delay is computed using Eq. (4) while the total dynamic power dissipation is given by

$$P_d = p + \alpha c V_{DD}^2 f \tag{10}$$

3.0 RESULTS AND DISCUSSION

3.1 Verification

To verify the correctness of the proposed iterative dynamic power computation scheme, the power dissipation computed by the proposed formulation is compared with the closed-form solution formulated in [5]. In this verification, the following wire and buffer parameters were used; r_w = 37.5 Ω/segment, c_w = 0.1026 pF/segment (where 1 segment = 200 μm), c_b = 0.0234 pF, r_b = 180 Ω, d_b = 36.4 ps, V_{DD} = 1 V, f = 2 GHz and α = 0.15 [5]. The same model setup as in [5] were used, where the source of the wire must be a buffer and the load capacitance is equal to the buffer input capacitance. Without loss of generality, the computations were performed on two-pin net as shown in Figure 5a.

In Figure 5a, the interconnect wire is equally segmented into seven segments represented in 1D graph where node 1 and node 8 are the source node and the sink node respectively. By using Eq. (6), the dynamic power dissipation of the interconnect in Figure 5a is therefore

$$P_d = 0.15 \cdot \left(\frac{102.6\,\text{fF}}{200\,\mu\text{m}} \cdot 1400\,\mu\text{m} + 3 \cdot (23.4\,\text{fF} + 202.2\,\text{fF}) \right) \cdot (1\,\text{V})^2 \cdot 2\,\text{GHz} = 0.419\,\text{mW} \tag{11}$$

Figure 5b shows the computation example of the proposed dynamic power computation scheme. The iterative computations start from node 8 (sink) toward node 1 (source). Recall that in this example, the sink load capacitance C_L is equal to the buffer input capacitance c_b. Therefore, the initial capacitance c at node 8 is 0.0234 pF and power p is 0 W. The next three path expansions (to nodes 7, 6 and 5) are wire expansions. Therefore, Eq. (1) and (7) are applied to compute the downstream capacitance c' and the downstream power p' (note that the value of delay is not shown in this example). The values of c and p at nodes 7, 6 and 5 are shown in Figure 5b. The expansion from node 5 to node 4 is the wire expansion terminated by a buffer (a buffer is inserted at node 5). Hence, Eq. (2) and (8) are applied. The computations return c = 0.126 pF and p = 0.16 mW at node 4. In the illustration, buffers are also inserted at nodes 3 and 1. At node 1, the total dynamic power dissipation, P_d for this interconnect is 0.419 mW. As shown in the computation, the total dynamic power dissipation computed by the new formulation is identical as the computation using the closed-form solution in [5].

The formulation had been incorporated into an improved van Ginneken buffer insertion algorithm, implemented in C programming language. The code was tested on many interconnect topologies and it produces the same dynamic power solution with the closed-form solution besides delay computation (Table 1). This proves that the proposed formulation can be integrated into dynamic programming buffer insertion algorithm.

3.2 Computation for Multiple Buffer Types

As stated earlier, the limitation of [5] and other closed-form solutions is that the closed-form solution can only compute power dissipation for one buffer type at a time. In other words, the formulation cannot handle multiple buffer types, i.e. when there are more than one buffer in the buffer library. Since the computation is done one segment at a time, the new formulation can handle any number of buffer types. It still applies Eq. (2) and (8), but it simply uses parameters suitable for each buffer type. Figure 6a shows the interconnect wire with two types of buffer inserted. The buffers are still at the same locations as in Figure 5a except that the buffer at node 3 is a buffer type 2 with the following parameters; c_b = 0.0117 pF, r_b = 360 Ω and d_b = 36.4 ps. The upstream computations are shown in Figure 6b. The computation returns total dynamic power dissipation, P_d at source of 0.385 mW. The computation proves that the proposed formulation is very useful in today's buffer insertion algorithm where multiple buffer types are prevalent and necessary. Other verification tests are shown in Table 1.

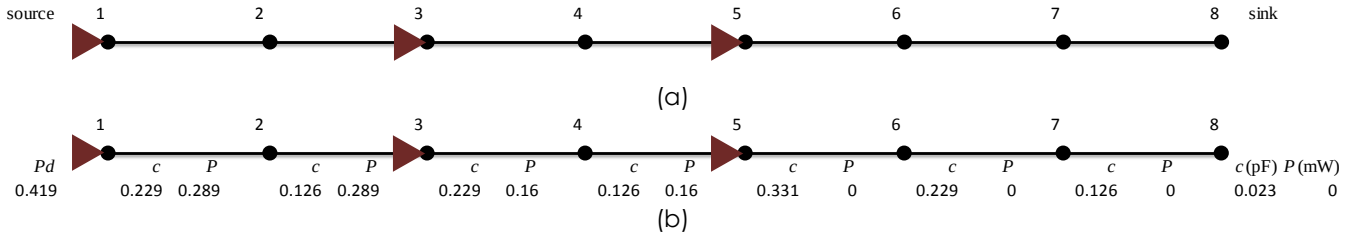


Figure 5 Illustration of the iterative power computation (a) sample net (b) upstream computation

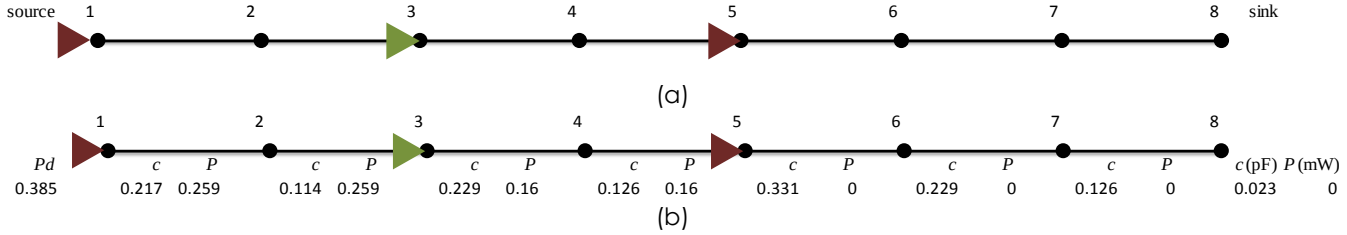


Figure 6 Illustration of the iterative power computation for multi buffer types (a) sample net (b) upstream computation

Table 1 Comparison between the proposed formulation and the closed-form solution [5]

Net	Proposed formulation		Number of inserted buffers	Closed-form solution [5]		Number of inserted buffers
	Delay (ps)	Power (mW)		Delay (ps)	Power (mW)	
In1	260.7	0.4	1	260.7	0.4	1
In2	482.8	0.6	2	482.8	0.6	2
In3	519.4	0.7	2	519.4	0.7	2
In4	743.6	1	4	743.6	1	4
In5	929.1	1.2	5	929.1	1.2	5

3.3 Test for Delay-Power Constraint Optimization

In this test, the proposed formulation is implemented in buffer insertion algorithm with delay-power constraint optimization. The algorithm is based on van Ginneken algorithm with the ability to satisfy different delay and power constraints. Tables 2 and 3 show the test results for optimization in two-pin nets and multi-pin nets (1 source, 4 sinks) respectively. In Table 2, for circuit In6, when the delay constraint is tight (1600 ps) and the power constraint is loose (1.7 mW), the algorithm inserts three buffers in order to satisfy the delay constraint. However, when the power constraint is tight (1.2 mW), the algorithm inserts only one buffer, resulting in more delay. The same effects are observed in nets In7 (Table 2) and In8 (Table 3).

Table 2 Test for delay-power constraint optimization in 2-pin nets

Net	Delay constraint (ps)	Power constraint (mW)	Results		Number of inserted buffers
			Delay (ps)	Power (mW)	
In6	1600	1.7	1331.6	1.4	3
	1900	1.2	1791.8	1.2	1
In7	1000	2	987.3	1.3	4
	1300	1	1246.7	1	1

Table 3 Test for delay-power constraint optimization in multi-pin nets

Net	Number of sinks	Delay constraint (ps)	Power constraint (mW)	Results		Number of inserted buffers
				Delay (ps)	Power (mW)	
In8	4	8500	30	7920	25	15
		8500	20	8211	17	8

4.0 CONCLUSION

A formulation to compute dynamic power consumption of buffers in dynamic programming framework for van Ginneken style buffer insertion algorithm is described. The proposed formulation is validated by comparing its computation result with the closed-form solution in [5]. The results show that the new formulation is correct and can be used in van Ginneken's buffer insertion algorithm with multiple buffer types. The implementation of the proposed algorithm for buffer insertion algorithm with delay-power constraint also shows that the proposed formulation is very useful for buffer insertion algorithm with multi-constraint optimization.

Acknowledgement

This work is supported by Higher Education Department, Ministry of Education Malaysia and Universiti Tun Hussein Onn Malaysia (UTHM).

References

[1] ITRS. 2013 International Technology Roadmap for Semiconductor: Interconnect. Available at: Http://www.itrs.net

[2] L. P. P. P. van Ginneken.1990. Buffer Placement In Distributed RC-Tree Networks For Minimal Elmore Delay. *Proc. Int. Symp. Circuits and Systems*. New Orleans, LA. 865-868.

[3] A. Nalamalpu and W. Burleson. 2001. A Practical Approach To DSM Repeater Insertion: Satisfying Delay Constraints While Minimizing Area And Power. *IEEE Con. ASIC/SOC*. Arlington, VA. 152-156.

[4] R. Li, D. Zhou, J. Liu, and X. Zeng. 2005. Power-Optimal Simultaneous Buffer Insertion/Sizing And Wire Sizing For Two-Pin Nets. *IEEE Trans. Comput. Des. Integr. Circuits Syst.* 24(12): 1915-1924.

[5] A. Narasimhan and R. Sridhar. 2010. Variability Aware Low-Power Delay Optimal Buffer Insertion For Global Interconnects. *IEEE Trans. Circuits Syst. I*. 57(12): 3055-3063.

[6] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. 2009. *Introduction to Algorithms*. 3rd ed. Boston, MA: McGraw-Hill.

[7] J. Lillis, C. K. Cheng, and T. T. Y. Lin. 1996. Optimal Wire Sizing And Buffer Insertion For Low Power And A Generalized Delay Model. *IEEE J. Solid-State Circuits*. 31(3): 437-447.

[8] L. Xun, P. Yuantao, and M. Papaefthymiou. 2005. RIP: An Efficient Hybrid Repeater Insertion Scheme for Low Power. *IEEE Design, Automation and Test*. 1330-1335.

[9] T. Murgan, O. Mitea, S. Pandey, P. Bacinschi, and M. Glesner. 2006. Simultaneous Placement and Buffer Planning for Reduction of Power Consumption in Interconnects and Repeaters. *IFIP Int. Conf. Very Large Scale Integr.* Nice. 302-307.

[10] C. J. Alpert, D. P. Mehta, and S. S. Sapatnekar. 2009. *Handbook Of Algorithms For Physical Design Automation*. Auerbach Publications.

[11] W. Elmore. 1948. The Transient Response Of Damped Linear Networks With Particular Regard To Wideband Amplifiers. *J. Appl. Phys.* 19(1): 55-63.

[12] N. Ekekwe. 2010. Power Dissipation And Interconnect Noise Challenges In Nanometer CMOS Technologies. *IEEE Potentials*. 29(3): 26-31.

[13] A. P. Chandrakasan, S. Sheng, and R. W. Brodersen. 1992. Low-power CMOS Digital Design. *IEEE J. Solid-State Circuits.* 27(4): 473-484.

[14] K. Banerjee and A. Mehrotra. 2002. A Power-Optimal Repeater Insertion Methodology For Global Interconnects In Nanometer Designs. *IEEE Trans. Electron Devices.* 49(11): 2001-2007.